POLITECNICO DI TORINO
Facoltà di Ingegneria dell'Informazione
Corso di Laurea in Ingegneria delle Telecomunicazioni

Tesi di Laurea

Studio ed implementazione di applicativi per la produzione di materiale multimediale distribuibile tramite Internet

Candidato:

Paolo CRAVERO

Relatori:

Prof. Matteo SONZA REORDA

Prof. Fulvio CORNO

Settembre 2001

Indice

Sommario 3
1 Introduzione 7
1.1 Obiettivo 7
1.2 Motivazione 7
1.3 Soluzione 7
1.4 Struttura 8
2 Ambiente operativo 9
2.1 Procedimento e applicazioni 9
2.1.1 Interfaccia utente 9
2.1.2 Registrazione di una lezione 11
2.1.3 Post-produzione 13
2.2 Struttura hardware e software 16
2.2.1 Interfaccia utente 16
2.2.2 Produzione 17
2.2.3 Post-produzione 17
2.2.4 Alternative commerciali 19
3 Software per la sincronizzazione 23
3.1 Identificazione problema 23
3.2 Procedimento logico 24
3.2.1 Specifiche 24
3.2.2 Esempi reali 24
3.2.2.1 Teletext 24
3.2.2.2 R.D.S. 25
3.2.2.3 C.T.C.S.S. 25
3.2.3 Psicoacustica 27
3.2.4 Steganografia 29
3.2.5 Larghezze di banda 29
3.2.6 Soluzione proposta 30
3.2.6.1 Premesse 30
3.2.6.2 Conclusioni 31
3.3 Generazione segnale di sincronia 33
3.4 Riconoscimento segnale di sincronia 38
3.4.1 Analisi spettrale 38
3.4.1.1 Analisi hardware 39
3.4.1.2 Analisi software 40
3.4.1.3 Windowing 40
3.4.1.4 Procedure software 42
3.4.2 Algoritmo di ricerca 42
4 Applicativi software 45
4.1 Applicativo software generico 45
4.1.1 Temporizzazioni 47
4.1.2 Processi concorrenti 50
4.1.3 Linux Open Sound System 52
4.2 Applicativo software specifico 54
4.2.1 Codifica AVI 54
4.2.2 Librerie MS AVIFile 56
5 Analisi dei risultati 63

5.1 Influenza sul prodotto finito 63
5.2 Parametri di ricerca 64
5.3 Fattore economico 67
5.4 Prestazioni 68
5.4.1 Applicativo generico 68
5.4.2 Applicativo specifico 69
6 Conclusioni 71
6.1 Sviluppi futuri 71
Appendice A Sony 9-Pin Remote Protocol 73
Appendice B Strutture dati AVI 103
B.1 AVISTREAMINFO 103
B.2 WAVEFORMATEX 106
Bibliografia 109

Sommario

Questo lavoro prende in considerazione il caso di lezioni e presentazioni da distribuire sotto forma di flussi informativi multimediali.

Le tecnologie disponibili consentono di inviare a distanza contenuti multimediali, costituiti da flussi audiovisivi, informazioni testuali e grafiche sincroni tra loro.

Tali flussi possono essere distribuiti in tempo reale o a richiesta (*real-time* o *on-demand*) sui canali trasmissivi di tipo Internet; così come possono essere registrati su CD-ROM per molteplici riproduzioni locali e personali su personal computer.

Nel caso del CD-ROM si aggiunge inoltre il fattore "interattività" che consente ai fruitori del materiale multimediale di spostarsi liberamente all'interno dei contenuti, e altrettanto facilmente raggiungere informazioni disponibili *online*, quindi sempre aggiornate.

La non deteriorabilità e il volume fisico richiesto per la conservazione delle lezioni multimediali a confronto con le normali videocassette VHS, inducono ad ipotizzare un futuro in cui le informazioni saranno archiviate e spostate solamente in formato numerico. Al contrario, le procedure necessarie alla produzione di materiale multimediale, con qualità audiovisiva paragonabile se non migliore di quella offerta da una videocassetta, comportano ancora una serie di passaggi obbligati ed un notevole impiego di risorse umane.

Il lavoro di questa tesi propone una soluzione al problema della sincronizzazione di immagini statiche con il flusso audiovisivo nella produzione di lezioni multimediali. L'insieme di procedure atte alla produzione di questo tipo di materiale è già consolidato (registrazione e post-produzione), così come la struttura dell'interfaccia utente del prodotto finito che permette di riprodurre su schermo quanto effettivamente avviene in una lezione dal vivo. La soluzione deve necessariamente inserirsi nelle sequenze operative già in atto, sostituendo un compito ripetitivo finora svolto da una persona.

L'obiettivo di automatizzare, e quindi velocizzare, la ricerca degli istanti in cui il relatore cambia il riferimento grafico (diapositiva computerizzata, *slide*) è stato raggiunto in due passi. Nella registrazione di audio e video (mezzobusto docente) su supporto ad alta qualità BetaCam viene affiancato un segnale sincrono con l'aggiornamento della diapositiva. Al fine di non introdurre ulteriore hardware nel sistema produttivo, le informazioni di sincronia sono inserite nella traccia audio della registrazione: audio, video e sincronismo sono sullo stesso supporto. In questo modo

si minimizza inoltre il rischio di smarrimento delle informazioni di sincronia, in quanto risiedono sullo stesso supporto fisico della lezione registrata.

Per non interrompere la continuità sonora della lezione, la segnalazione binaria con significato di "cambio diapositiva" si realizza con l'inserimento di un tono sincrono di durata e frequenza prestabilite. Sfruttando contemporaneamente l'estesa banda passante dei sistemi di registrazione audio al confronto della ristretta banda effettivamente udibile da un orecchio umano, la frequenza del tono di sincronia viene scelta al di sotto della banda udibile (100 Hz): si realizza così una segnalazione associata al canale audio assolutamente trasparente all'utente finale. L'inserimento di informazioni senza alterare l'intelligibilità del messaggio ospitante si chiama *steganografia*.

Ad ulteriore supporto della soluzione presentata va considerato che la compressione MPEG attuata sulla traccia audio della lezione si basa sulle conclusioni della *psicoacustica*, ed è quindi propensa ad eliminare totalmente tutto quanto non udibile da un essere umano, cioè a cancellare il segnale di sincronia a quel punto già utilizzato nelle procedure di post-produzione.

Il recupero della sincronia avviene, infatti, durante la serie di operazioni che si eseguono tra la registrazione e il prodotto finito. Questa serie di operazioni rappresenta la post-produzione.

La ricerca del segnale di sincronia avviene analizzando la traccia audio della lezione con una trasformata discreta di Fourier (DTFT). Con un algoritmo (sviluppato *ad-hoc* dallo scrivente) vengono osservate le variazioni di ampiezza nella componente spettrale corrispondente alla frequenza del tono subsonico utilizzato. Quando l'ampiezza supera il livello di soglia per un tempo prestabilito l'applicativo valida il segnale di sincronia. Il livello di soglia è definito dall'operatore prima della ricerca.

L'*output* del procedimento di ricerca è un file testuale contenente le *azioni* valide da associare alla codifica finale del flusso audio/video nel formato ASF, che si effettua al passo successivo della post-produzione. Tali *azioni* vengono eseguite durante la riproduzione del prodotto finito, all'interno di una finestra del *browser* Web, e comandano il cambio dell'immagine computerizzata statica equivalente alla *slide* originale.

Lo sviluppo di software atto alla ricerca e al riconoscimento del tono di sincronia ha portato a due applicativi che, sostanzialmente equivalenti nel raggiungere entrambi l'obiettivo, operano in modo diverso e su sistemi operativi differenti.

Il primo è caratterizzato dal dialogo con il videoregistratore tramite protocollo seriale standard. La lezione viene riprodotta direttamente dalla cassetta BetaCam. La

sincronia viene recuperata mediante acquisizione ed elaborazione in tempo reale della traccia audio. Il codice sviluppato funziona su sistemi operativi Linux compatibili e non necessita dell'intervento di un operatore durante la ricerca.

Il secondo si distingue per l'utilizzo come *input* dei file AVI risultanti dall'acquisizione ad alta qualità del flusso audiovisivo. La numerizzazione ad alta qualità della lezione caratterizza il primo passo della post-produzione. Il compito di un operatore è limitato all'inserimento dei nomi dei file di ingresso. L'applicativo opera su elaboratore con sistema operativo Microsoft Windows.

I risultati ottenuti verificando operativamente entrambi gli applicativi hanno provato che la tecnica steganografica di registrazione della sincronizzazione garantisce caratteristiche di affidabilità e precisione, tali da soddisfare i requisiti necessari per l'automatizzazione di un passo della post-produzione di lezioni multimediali.

1 Introduzione

1.1 Obiettivo

Il presente lavoro si propone di apportare maggiore automatizzazione nel processo di post-produzione di contenuti multimediali (audio/video e diapositive sincroni), con il preciso intento di ridurre le ore-uomo necessarie ad ultimare il singolo prodotto. La collaborazione con il **Ce**ntro per i Servizi **Te**ledidattici e **M**ultimediali del Politecnico di Torino (Ce.Te.M.) ha permesso di quantificare i risultati ottenuti.

1.2 Motivazione

Al termine della registrazione di ogni lezione destinata ad essere distribuita agli studenti su un qualunque supporto informatico, vi è un lungo lavoro di montaggio, come per le normali registrazioni video, a cui si aggiunge la necessità di integrare delle immagini statiche e di sincronizzare queste ultime con il flusso audio/video. Avendo il Ce.Te.M già definito tutti i passi della produzione delle menzionate lezioni multimediali così come la disposizione grafica finale, è necessario intervenire su tale processo nel modo meno invasivo possibile, rispettando contemporaneamente i vincoli di:

- non dover investire in formazione del personale che si occupa della produzione
- non dover investire in nuovo hardware
- non cambiare le abitudini già consolidate degli utenti finali.

1.3 Soluzione

Considerata la limitata capacità uditiva dell'orecchio umano rispetto a quanto offerto dalla tecnologia esistente per la registrazione e riproduzione di suoni, si è pensato di inserire un tono a frequenza subsonica (di lunghezza e frequenza prestabilite e note) sulla traccia audio della registrazione originale, sincrono con l'inizio del tempo di proiezione di ogni diapositiva. In fase di post-produzione si effettua via software la ricerca del tono e si genera, automaticamente, l'elenco delle coppie tempo/diapositiva da trasferire al passo successivo del processo produttivo.

1.4 Struttura

Nel presente documento si analizza innanzi tutto (Capitolo 2) l'ambiente operativo in cui deve integrarsi la soluzione proposta, il quale comprende:

- un insieme di procedure di post-produzione
- una infrastruttura preesistente hardware e software.

Segue una descrizione dettagliata delle due soluzioni software (Capitolo 3) a cui si è giunti durante il lavoro di sviluppo, le quali perseguono lo stesso obiettivo ma operano in modo ed in ambienti diversi (Capitolo 4).

Una valutazione quantitativa dei benefici apportati da questo prodotto (Capitolo 5) precede le conclusioni (Capitolo 6). Nelle appendici si descrivono il protocollo SONY per la comunicazione con videoregistratori BetaCam (Appendice A) e due strutture dati per la gestione di file AVI (Appendici B).

2 Ambiente operativo

Questo capitolo si propone di offrire una panoramica sulla infrastruttura necessaria alla produzione di contenuti multimediali da distribuire su supporto informatico, come CD-ROM o via Internet. Il riferimento a *lezioni multimediali* non limita le considerazioni che seguono al solo ambito della istruzione a distanza. Le soluzioni descritte nei capitoli successivi trovano applicazione ovunque sia necessario sincronizzare flussi multimediali eterogenei.

2.1 Procedimento e applicazioni

2.1.1 Interfaccia utente

Al fine di entrare nell'ottica di quanto verrà descritto più avanti si analizzano qui di seguito la struttura del prodotto finito, nonché le differenze tra una lezione dal vivo ed una su computer.

Il contenuto tipico di una lezione universitaria, usando come riferimento le lezioni di Ingegneria che hanno accompagnato l'autore nel suo percorso formativo, prevede una platea di studenti i quali guardano e ascoltano un relatore descrivere delle diapositive proiettate dalla lavagna luminosa (o da un computer). Ogni lezione dura circa 50 minuti. Questa procedura di apprendimento è ormai consolidata nelle abitudini di ogni studente che percorre il suo iter universitario (ma anche di coloro che hanno dovuto seguire dei corsi di qualunque tipo in ambiente extra-universitario), ed è dunque necessario che una lezione di tipo multimediale mantenga quanto più possibile la stessa struttura.

Dovendo registrare una lezione per riprodurla in tempi e luoghi differenti si devono allora identificare e riportare tre flussi di informazione: audio (voce del docente), video (figura del docente), diapositive (rappresentazione grafica della struttura del discorso).

Il flusso audio è ovviamente quello a maggiore contenuto informativo, e deve essere riprodotto con la maggiore fedeltà possibile. Esso viene solitamente affiancato ad un flusso video con la figura a mezzobusto del docente in movimento. Sebbene quest'ultimo flusso di dati non contenga informazioni rilevanti per lo studente e richieda una notevole banda trasmissiva, è stato provato che l'immagine in movimento del docente aiuta a mantenere la concentrazione sul contenuto del parlato. Infine vi è l'insieme delle informazioni testuali rappresentate nelle diapositive che possono essere assimilate ad immagini statiche. Mentre l'audio ed il video possono

essere facilmente combinati in un unico flusso informativo, e quindi riprodotti nello stesso momento, la parte testuale non può essere ivi inclusa: ciò comporterebbe un continuo cambio di inquadratura tra il docente e la diapositiva, perdendo la funzione positiva ipotizzata per la presenza continua del docente sullo schermo.

E' inoltre risaputo che non esiste un tempo di esposizione delle diapositive che accontenti tutta la platea: un tempo di esposizione troppo corto costringe lo studente ad un salto indietro nella lezione registrata con conseguente inevitabile perdita di continuità e concentrazione. Al contrario un tempo di esposizione troppo lungo rilassa l'attenzione dell'ascoltatore, la quale però è mantenuta viva dalla visione della figura in movimento del docente.

Figura 2.1 Interfaccia per l'utente di una lezione multimediale adottata dal Ce.Te.M.

Un esempio di interfaccia utente è quella sviluppata dal Ce.Te.M. (riportata in Figura 2.1). Essa tende a massimizzare l'utilizzo delle risorse informatiche a disposizione dello studente ed il suo livello di attenzione, visualizzando in contemporanea la figura del docente in movimento (a bassa risoluzione, per limitare le dimensioni del relativo file e le risorse necessarie alla riproduzione) e la diapositiva relativa al parlato (a centro schermo per garantire la migliore leggibilità). L'intera

schermata con la lezione viene visualizzata da un *browser* Web dove il lettore di filmati multimediali (in alto a sinistra) controlla anche il cambio delle diapositive statiche (parte centrale) sincronizzandole con il parlato. Per una descrizione dettagliata dei requisiti software si veda *2.2.1 Interfaccia utente*.

La lezione multimediale così come viene realizzata al momento è limitata alla distribuzione su CD-ROM. Peraltro non vi è bisogno di alcuna revisione alle procedure di produzione se si desidera renderla disponibile ad un più vasto pubblico tramite pubblicazione su Internet. Si noti infine come lo stoccaggio di un CD-ROM richieda minor volume rispetto ad una videocassetta, e come la sua qualità di riproduzione non degradi con l'utilizzo ripetuto rispetto a ciò che avviene con una videocassetta.

2.1.2 Registrazione di una lezione

Alla luce delle considerazioni di cui sopra, la registrazione di una lezione deve soddisfare alcuni requisiti di qualità e flessibilità. E' infatti necessario che i tre flussi possano essere riprodotti in modo indipendente in un secondo tempo, per essere codificati numericamente con qualità differenti: mentre l'audio deve essere il più fedele possibile (qualità "CD"), il video sarà rimpicciolito e degradato al termine del processo di produzione. In ogni caso, per avere una buona qualità finale, è necessario partire da audio e video registrati in altissima definizione, cioè utilizzando attrezzature professionali e standard BetaCam[1].

Le diapositive proiettate sullo schermo del computer al tavolo del docente rimangono in forma numerica, come immagini statiche in formato GIF o JPEG[2]. Non vengono pertanto riprese dalle cineprese e registrate su nastro: tale operazione produrrebbe solo una degradazione della loro qualità ed una complicazione inutile per la regia della registrazione.

Si immagini quindi uno studio di registrazione composto da due sale: regia e camera insonorizzata. Nella camera insonorizzata si trova il docente, seduto ad un tavolo provvisto di computer (monitor, tastiera e mouse visibili nelle riprese), e due

---

[1] BetaCam è un formato standard di registrazione video ad alta qualità basato sulle componenti in cui si può scomporre un segnale video: 1 di luminanza e 2 di crominanza. I tre segnali sono registrati in modo indipendente per minimizzare la perdita di segnale durante la registrazione/riproduzione.

[2] GIF: Graphical Interchange Format. JPEG: Joint Picture Expert Group. Due tra i più diffusi formati di codifica per computer di immagini statiche.

addetti alle riprese (*cameraman*), uno per ciascuna cinepresa. In sala regia vi sono uno o due registi che controllano i livelli di registrazione audio/video e comandano gli operatori di ripresa cambiando le inquadrature quando opportuno nonché prendono nota dei tempi di inizio e fine registrazione.

Ogni lezione, infatti, viene suddivisa in blocchi di alcuni minuti corrispondenti agli argomenti trattati dal docente, giungendo così ad un prodotto finito composto da tanti argomenti elementari riproducibili singolarmente e in tempi diversi. Tra le due sale scorrono i cavi che trasportano i segnali video e audio ai rispettivi miscelatori, e la registrazione su cassetta Betacam [1] contiene una traccia video ed una audio stereo (entrambe analogiche).

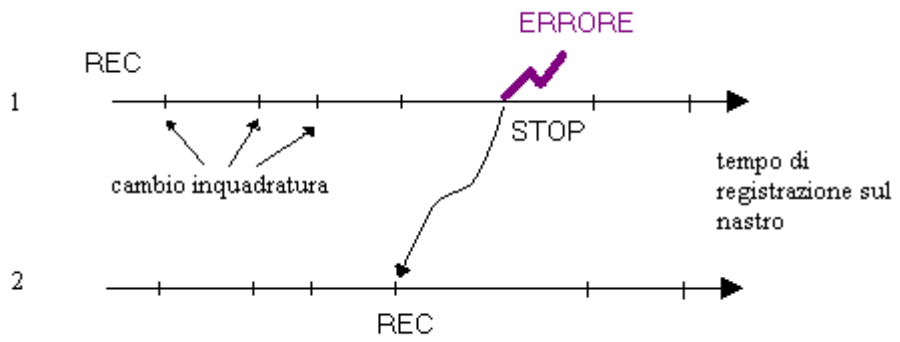

Figura 2.2 Correzione di un errore del docente in fase di registrazione. Nel momento in cui il docente dichiara l'errore (1), si riavvolge fino al cambio di inquadratura precedente e si riprende la registrazione (2).

In caso di errori del docente (o in ogni caso quando ciò fosse necessario) gli addetti alla regia provvedono a riavvolgere il nastro ed a riprendere le registrazioni in modo tale da sovrascrivere la parte errata, solitamente in concomitanza con un cambio di inquadratura: non sarà così necessario un lavoro di montaggio video a posteriori per eliminare l'errore (Figura 2.2).

Al termine della registrazione di una lezione sono disponibili:

- Videocassetta Betacam contenente la traccia audio (voce del docente mescolata con eventuali musiche e suoni) sincrona con la traccia video (inquadratura a mezzobusto in movimento)

- Elenco dei tempi di inizio e fine di ogni blocco della lezione (*timecode*[3]) in minuti, ore, secondi e numero di *frame*
- Insieme delle diapositive utilizzate nel corso della lezione, su supporto informatico

2.1.3 Post-produzione

In questa fase si eseguono una serie di procedure, ripetitive e parzialmente automatizzate, che permettono di giungere al prodotto finito partendo dai tre punti elencati precedentemente***. Una di queste procedure è stata automatizzata nell'ambito di questa tesi.***

La post-produzione necessita di un videoregistratore Betacam interconnesso ad un computer con adeguata potenza di calcolo e dotato di scheda di acquisizione audio/video professionale. Per una descrizione dettagliata dei sistemi hardware e software utilizzati in post-produzione si veda *2.2.3 Post-produzione*.

La prima fase della post-produzione consiste nel trasferire su computer i blocchi della lezione, cioè convertirli in formato numerico, mantenendo la maggiore qualità possibile.

```
Lezione    blocco1      01:02:00:00    01:06:55:10
Lezione    blocco2      01:07:30:02    01:15:10:00
Lezione    blocco3      01:16:12:00    01:21:35:12
```

[3] Le registrazioni Betacam scrivono sul nastro anche l'informazione relativa al trascorrere del tempo di registrazione, composta da ore, minuti, secondi e numero del frame (30 frame al secondo). Tale informazione non dipende da un contatore autonomo come nei videoregistratori amatoriali, ma può essere visualizzata univocamente in qualunque punto del nastro ci si trovi.

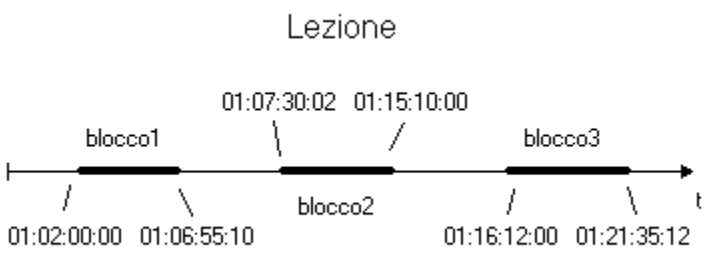

Figura 2.3 Rappresentazione sull'asse temporale dei timecode dei blocchi appartenenti ad una lezione

Si parte dalla videocassetta e da un file contenente l'elenco dei tempi di inizio e fine di ogni blocco (*timecode* di IN e OUT) per giungere ad un insieme di file AVI[4] con qualità video di 6 Mbit/s e audio campionato a 44.1 kHz, codifica PCM[5] a 16bit, monoaurale (si registrano entrambi i canali destro e sinistro, con lo stesso contenuto). L'acquisizione avviene in sequenza e in modo automatico da parte un software atto al montaggio audio/video (*Adobe Premiere*), il quale dialoga con il videoregistratore Betacam. Dovendo leggere il nastro alla normale velocità di riproduzione, questa operazione richiede un tempo pari alla durata della lezione, ma non necessita della presenza di un operatore.

I risultanti file devono invece essere verificati da un operatore, il quale agisce al fine di tagliare le parti non necessarie all'inizio e alla fine di ciascun file. Allo stesso tempo l'operatore deve procedere a recuperare l'informazione di "cambio diapositiva" ed annotare i relativi tempi con riferimento all'inizio del blocco, per sincronizzare i due flussi (audio/video e diapositiva) nel prodotto finito.

Prima del presente lavoro sperimentale, quest'ultima operazione richiedeva che un operatore guardasse tutti i blocchi della lezione registrata alla ricerca del cambio diapositiva, corrispondente al movimento della mano del docente sul mouse o sulla tastiera. Ora invece è possibile immagazzinare permanentemente questa informazione in fase di registrazione direttamente sul nastro Betacam, per poi recuperarla automaticamente durante la post-produzione con un apposito software come descritto

---

[4] Audio Video Interleave. E' un caso particolare di RIFF (Resource Interchange File Format). Il formato AVI è stato definito da Microsoft ed è il formato più comune per dati audio/video su Personal Computer. AVI è uno standard de-facto.

[5] Pulse Coded Modulation: si registra il valore assoluto di ogni campione del segnale analogico, con risoluzione in bit/campione predeterminata (di solito 8 o 16 bit/campione).

nel capitolo seguente. Segue un esempio di file di *output* del procedimento automatico di ricerca della sincronia, nonché di *input* al passo successivo:

```
;Properties
Title:
Author: Paolo Cravero
Copyright:
Description: Test Tesi
Rating:

;Script Commands
start_script_table
00:00:00.1 URL diapositiva1.jpg
00:00:14.5 URL diapositiva2.jpg
00:01:30.2 URL diapositiva3.jpg
end_script_table
```

Infine ogni singolo blocco di lezione e la corrispondente lista dei tempi di cambio diapositiva vengono combinati e codificati in un singolo file ASF[6]. Durante questo passo (*rendering*) il video e l'audio sono ricodificati per un *bit-rate* più basso, quindi per una qualità di riproduzione minore. I tempi di conversione per l'insieme di blocchi di una lezione di 45 minuti si attestano sulle 12 ore con pieno utilizzo della CPU del computer interessato.

Le diapositive, estratte dalla presentazione *PowerPoint* in un formato grafico supportato dal browser Web (GIF o JPEG) verranno copiate assieme al video ASF sul supporto finale, secondo una struttura definita. Allo stesso tempo si rende necessario adattare al contesto della lezione (nome docente, nome corso, indice blocchi, …), manualmente e *una tantum*, i pochi file HTML che compongono l'interfaccia utente del prodotto finito.

---

6 Advanced Streaming Format: formato di file per la riproduzione di contenuti multimediali sincronizzati. Supporta la distribuzione di contenuti multimediali generati in diretta o su richiesta (*on-demand*). Il formato dei dati in un file ASF è adatto sia allo streaming sia alla riproduzione locale. [2]

2.2 Struttura hardware e software

2.2.1 Interfaccia utente

Le tecnologie software disponibili attualmente limitano il campo di utilizzo delle lezioni multimediali al solo sistema operativo Microsoft Windows. Si tratta di un compromesso sul prodotto finito atto a raggiungere il maggior numero di utenti e allo stesso tempo soddisfare gli obiettivi di una lezione multimediale.

Sebbene non sia il sistema operativo in sé ad essere limitativo, lo è invece il software che permette di visualizzare il formato ASF. Il visualizzatore ASF, *player*, al momento non è disponibile per piattaforme software diverse da Windows. "Advanced Streaming Format" è infatti visualizzabile solo tramite Windows Media Player versione 6.4 o successiva. I *browser* supportati sono sia *Netscape Navigator* sia *Internet Explorer*, nonostante le particolari API[7] utilizzate per aggiornare le diapositive siano differenti per i due software (e anche per versioni differenti dello stesso *browser*).

Windows Media Player interagisce perfettamente con un *browser*: un filmato in formato ASF se visualizzato all'interno di una pagina Web può inviare comandi al *browser*. Si pensi ad una pagina con struttura "a cornici" (equivalente al termine inglese *frames*), in tal caso il filmato multimediale può indicare al *browser*, tramite il *player*, di caricare un URL[8] all'interno di un qualsiasi *frame* appartenente alla stessa pagina.

Dal punto di vista hardware, una qualunque configurazione "PC multimediale" è sufficiente a visualizzare le lezioni, ferma restando una capacità di calcolo minima equiparabile ad un processore Intel Pentium 200 MHz e una quantità minima di memoria di sistema pari a 64 MByte.

---

[7] Application Programmer Interface: un insieme documentato di chiamate a funzione per interagire con software esistente (programmi, librerie, …).

[8] Uniform (o Universal) Resource Locator: una sequenza di caratteri che identifica univocamente un documento su Internet o sul disco locale (ipertesto, immagine, filmato, …). Specifica il protocollo da utilizzare per il trasferimento, il server e la porta a cui collegarsi, la directory e il file da trasferire (esempio: http://www.polito.it:80/index.html). [3]

2.2.2 Produzione

Le attrezzature necessarie alla produzione di una lezione multimediale non differiscono molto da quelle utilizzate per la realizzazione di un programma televisivo. Si veda un elenco delle principali apparecchiature a disposizione del Ce.Te.M.

Molteplici segnali prodotti nella sala di ripresa giungono ai miscelatori audio e video della regia tramite cavi schermati bilanciati:

- Audio prodotto dalla scheda audio del computer a disposizione del docente
- Audio prodotto dal microfono del docente
- Video prodotto dal computer a disposizione del docente (uscita della scheda video in formato VGA rimodulata in PAL da uno *Scan Converter*)
- Immagini riprese dalle due telecamere BetaCam
- Immagini riprese dalla telecamera statica a soffitto posizionata sopra il tavolo del docente (al momento non utilizzata per le lezioni multimediali)

In sala regia i segnali video possono essere visualizzati indipendentemente su monitor differenti, mentre l'insieme dei segnali audio viene abitualmente ascoltato dopo la relativa miscelazione, così come viene registrato su cassetta.

I cablaggi tra le due sale sono effettuati il modo permanente, e un vetro permette ai registi di comunicare, a gesti, con il docente. Operatori di ripresa e un regista sono invece collegati tramite interfono.

Nel caso del Ce.Te.M. la regia dispone di un registratore BetaCam *SONY PVW2800P*, un miscelatore audio *Spirit Live* a 8 piste stereo; un miscelatore video *Grass Valley GVG100CV* a 8 canali. Lo Scan Converter è un *SONY DSC1024G*.

2.2.3 Post-produzione

Tutta la sequenza di operazioni di post-produzione richiede un insieme di dispositivi più ristretto rispetto alla produzione, e può avvenire ovunque vi sia l'infrastruttura necessaria.

Dovendo acquisire audio e video su computer da cassetta BetaCam, sono necessari sia un videoregistratore BetaCam sia un computer con una scheda di acquisizione professionale tipo *Pinnacle Reeltime NITRO* o *Truevision TARGA 2000RTX*. Il computer, con sistema operativo Windows NT, ha una CPU Intel

Pentium III 450MHz, 512 Mbyte RAM, due dischi rigidi SCSI in configurazione RAID 0[9], connessione alla rete locale per la condivisione di risorse. Tra il videoregistratore e il computer vi sono i cavi per il trasporto di audio e video analogici. Elettricamente, i segnali non hanno un riferimento di massa comune, bensì viaggiano su cavi bilanciati.

Il computer, tramite il software *Adobe Premiere*, consente una gestione completa del videoregistratore BetaCam previa interconnessione tramite cavo di trasmissione bidirezionale seriale con standard RS-422[10]. Il protocollo di comunicazione seriale è stato introdotto dalla SONY ed è utilizzato come standard per la gestione remota di dispositivi BetaCam di tutte le case produttrici. Lo utilizzano infatti tutte le centraline di controllo dei registratori BetaCam, le quali funzionano con i prodotti di tutte le Case. Se il computer dispone solo di una porta con standard RS-232, è necessario un adattatore di livello esterno. Il protocollo seriale che ha velocità di 38400 bit al secondo e dati da 8 bit con parità pari, è documentato nella Appendice A.

Adobe Premiere gestisce la comunicazione sia con il terminale remoto (seriale) sia con la scheda di acquisizione. Quest'ultima è utilizzabile solo da pochissimi software disponibili sul mercato, e non esiste documentazione gratuita per sviluppare le proprie applicazioni. A *Premiere* si passa il file con i tempi dei blocchi (visto in 2.1.3 Post-produzione) in cui è suddivisa la lezione e automaticamente vengono generati tanti file AVI quanti sono i blocchi.

I risultanti AVI contengono una copia fedele del contenuto della cassetta BetaCam, ossia tutta la traccia audio e tutta la traccia video (incluse le linee normalmente non visibili sui televisori, cioè un riquadro video più alto e largo del normale).

Al passo successivo, oltre al recupero della sincronizzazione con le diapositive qui non considerato, l'operatore deve provvedere a ritagliare opportunamente il video, sia per quanto riguarda le dimensioni sia per inserire o rimuovere le sequenze che non

---

[9] Reduntant Array of Inexpensive Disks: è una tecnologia implementata sia via hardware sia via software (per Linux) che permette di scrivere la stessa informazione contemporaneamente su più dischi rigidi identici in modo da creare automaticamente ed istantaneamente una copia di ogni dato scritto. Si dimezza quindi la probabilità di perdere dei dati in caso di problemi all'hardware dei dischi. Il modo RAID 0 legge e scrive in parallelo su due dischi, senza aggiungere ridondanza ai dati. [5]

[10] Lo standard RS-422 definisce i livelli di tensione, le lunghezze e le velocità di trasmissione seriali per interfacce digitali bilanciate. [4]

appartengono al blocco della lezione. E' prassi comune aggiungere in testa ed in coda al filmato alcuni secondi di schermo nero, più eventuali effetti di evanescenza.

2.2.4 Alternative commerciali

Al momento in cui si scrive esiste un solo prodotto commerciale che permette di sincronizzare flussi audio/video con diapositive: *RealPresenter Plus* versione 8, prodotto dalla RealNetworks (www.real.com). RealPresenter Plus fa parte di una serie di programmi dedicati alla produzione e diffusione tramite Internet di oggetti multimediali. Appartengono allo stesso gruppo: *RealSystem Producer Plus* per la produzione di contenuti audio/video per Internet *streaming*, *RealSlideshow Plus* per l'integrazione di audio e immagini statiche, *RealSystem Server Plus* che si occupa dello *streaming* vero e proprio. Le applicazioni per *streaming* della RealNetworks usano gli standard proprietari denominati RealAudio (per il solo audio) e RealVideo (per audio, video e contenuti multimediali).

E' comunque possibile salvare su file e distribuire quest'ultimo senza passare necessariamente dal server di *streaming*, utilizzando, come per la soluzione proposta da Microsoft, il lettore dedicato per una riproduzione su macchine anche non connesse in rete (*RealPlayer*). Si tratta pertanto del maggiore concorrente al pacchetto di *streaming* multimediale di Microsoft (Windows Media), con cui condivide buona parte degli utenti.

Come si legge in "RealNetworks supera Microsoft nel numero di utilizzatori di lettori multimediali" [6], da un rapporto di Jupiter Media Matrix, nel novembre 2000 il 50% degli utenti americani di Internet usava una delle due tecnologie esposte sopra per visualizzare filmati online, con un leggero vantaggio del sistema della RealNetworks (28% contro il 22% di Microsoft sul totale degli utenti statunitensi di Internet). Sempre Jupiter Media Matrix nell'aprile 2001 conferma la differenza nel numero di utenti delle due tecnologie principali [7], ed evidenzia anche una notevole crescita degli utenti totali di Internet *streaming* negli ultimi mesi. Allo stesso tempo, però, la maggior parte degli utenti intervistati dichiara di cambiare spesso il player utilizzato, dimostrando così una scarsa fedeltà ad un determinato sistema di riproduzione multimediale. Ciò può significare che per canali a banda trasmissiva estremamente limitata quali sono i collegamenti telefonici, in confronto alle necessità delle applicazioni multimediali di alta qualità, le prestazioni qualitative offerte dai due software sono pressoché equivalenti.

Si tenti allora un confronto tra le due tecnologie evidenziandone i punti salienti a favore o sfavore, limitandosi all'ambito dello scopo di questa tesi.

Punti di forza	***Punti a sfavore***
• L'acquisizione, conversione e sincronizzazione di audio/video con le informazioni testuali avviene in un solo passaggio • Server di *streaming* integrato, ma per poche connessioni simultanee • Lettore disponibile gratuitamente (in versione base) per i sistemi operativi maggiormente diffusi • Protezione dei diritti d'autore delle informazioni testuali incluse nella presentazione	• Formato di *streaming* proprietario • Richiede MS PowerPoint 97 o 2000 • Non consente l'utilizzo di sistemi di acquisizione professionali • L'integrazione delle immagini nel file di *streaming* ne aumenta le dimensioni, richiedendo maggiore banda trasmissiva • Programmi di acquisizione, sviluppo e diffusione a pagamento • Ambiente di sviluppo solo per MS Windows • Codifiche con rapporti bit al secondo differenti, per accontentare tutte le utenze, richiedono file diversi (URL differente per ogni bit rate)

Tabella 2.1 Vantaggi e svantaggi di RealPresenter Plus

Dalla Tabella 2.1 si nota come a fronte della disponibilità del lettore RealAudio/RealVideo per più sistemi operativi, tutto il software nella versione non "base" sia a pagamento. Inoltre per l'applicazione che sta alla base di questo lavoro di ricerca, cioè l'integrazione audio/video con immagini statiche, la soluzione proposta dalla RealNetworks non sembra soddisfare i requisiti di versatilità, economicità e qualità che ci siamo prefissi: non permette infatti di utilizzare file AVI come sorgenti audio/video, ma si limita al video registrabile con una comune *webcam*[11].

[11] Si indica con il termine "webcam" una mini-telecamera da collegare ad un personal computer, tipicamente tramite interfaccia USB, utilizzabile per riprese a bassissima qualità da trasmettere direttamente su canali con limitata banda trasmissiva (56 kbit al secondo o minore).

Limitatamente all'ambito della produzione delle lezioni multimediali prevalgono gli aspetti negativi di RealPresenter: software a pagamento e impossibilità di utilizzare file AVI.

Punti di forza	***Punti a sfavore***
• Lettore perfettamente integrato con MS Windows, già incluso in quest'ultimo • Il prodotto finale (formato ASF) ha come punto di partenza un file in formato AVI, che può essere generato con attrezzature professionali • Codifiche con rapporti bit al secondo differenti, per raggiungere tutte le utenze, contenute in un singolo file (unico indirizzo per tutte le utenze) • Programmi di acquisizione, sviluppo e diffusione gratuiti • Possibilità di integrare nella presentazione qualunque ipertesto, non solo diapositive PowerPoint • Molteplicità di programmi per generare e maneggiare il prodotto intermedio (formato AVI) • Protezione del copyright per i flussi audio e video	• Ambiente di sviluppo disponibile solo per piattaforme MS Windows • Lettore per formato ASF disponibile solo per MS Windows • Nessuna protezione di copyright per le immagini inserite esternamente • Impossibile tornare al formato AVI dal formato ASF

Tabella 2.2 Vantaggi e svantaggi di Microsoft Media

La soluzione proposta da Microsoft (riassunta in Tabella 2.2) ha il vantaggio di disporre di molteplici ambienti di sviluppo (non obbligatoriamente legati ad essa da accordi commerciali), in virtù del fatto che tutto il lavoro di post-produzione che segue le registrazioni si effettua su un formato di file standard (AVI), per poi essere convertito nel prodotto finale adatto allo *streaming* (Advanced Streaming Format, ASF). Per contro vi è lo svantaggio, non piccolo, che l'utenza finale è costretta ad utilizzare *Windows Media Player* per visualizzare i contenuti multimediali ASF, ed è quindi limitata alle sole piattaforme su cui funzionano i sistemi operativi di Microsoft. Allo stesso tempo la protezione dei diritti d'autore offerta da Microsoft Windows Media si limita al fatto che la codifica nel formato ASF adottata da Microsoft non è

pubblica, e quindi nessun software può estrarre e copiare le informazioni audio e video ivi contenute.

La scelta delle tecnologie proposte da Microsoft come supporto per il prodotto finito di un centro di teledidattica è obiettivamente ed attualmente quella che offre maggiore versatilità ad un costo sensibilmente minore rispetto alla soluzione concorrente. Se si considera inoltre che gli utenti interessati ai contenuti multimediali disponibili su supporto informatico (CD-ROM o direttamente su Internet) non abbiano particolari preferenze sul formato utilizzato, un centro di produzione può scegliere il processo produttivo che maggiormente soddisfa alle specifiche tecniche di produzione (nel caso del Ce.Te.M.: Microsoft).

3 Software per la sincronizzazione

In questo capitolo si analizza il problema, riassumendo quando esposto nei capitoli precedenti, e in dettaglio le soluzioni a cui si è giunti.

3.1 Identificazione problema

Le tecnologie disponibili permettono ad un centro di produzione per contenuti multimediali di affidare la distribuzione del materiale didattico al solo supporto informatico, quale CD-ROM o Internet e abbandonare conseguentemente la videocassetta VHS.

Data la natura multimediale ed interattiva che si può imporre ad un prodotto da utilizzare su PC, ogni lezione visualizza nello stesso momento sia la figura del docente sia le diapositive (*slide*) da quest'ultimo utilizzate. Il flusso audio/video provvede a sincronizzare l'aggiornamento a video delle diapositive. Al fine di semplificare il processo produttivo, di mantenere limitati i requisiti hardware per l'utente finale e, non ultimo, di non dover gestire troppi file a contenuto multimediale (AVI), solo i flussi audio e video vengono registrati (BetaCam) e in un secondo tempo codificati secondo un formato di *streaming* (ASF). Le diapositive rimangono invece sotto forma di immagini statiche GIF o JPEG, non integrate all'interno del file di *streaming* vero e proprio.

Si manifesta la necessità di mantenere l'informazione relativa all'aggiornamento della *slide* cui il docente fa riferimento, durante il processo di registrazione (ad alta qualità con attrezzature professionali). Tale informazione è un valore temporale riferito all'inizio della registrazione, o ad una parte di essa qualora la lezione venga suddivisa in più blocchi. In assenza del segnale di sincronia video/diapositive, nella fase di post-produzione un operatore deve ripercorrere tutta la lezione (o i singoli blocchi) alla ricerca del momento in cui il docente ha comandato il cambio diapositiva mediante la pressione di un tasto sulla tastiera o del mouse. L'operazione di revisione della lezione richiede tempo e può produrre dei risultati non veritieri (in termini di valori temporali): infatti lo scarto tra il momento reale di aggiornamento della diapositiva e quello rilevato dall'operatore può ripercuotersi sul prodotto finito in modo fastidioso per l'utente.

L'innovazione consiste nell'introdurre una automatizzazione al processo di recupero dell'informazione temporale, al fine di ridurre il tempo totale di produzione di una lezione multimediale. Una qualunque soluzione deve integrarsi nella struttura e nelle procedure preesistenti, senza introdurre particolari complicazioni al personale

addetto alla produzione o modificare fisicamente le connessioni elettriche tra i vari dispositivi atti alla registrazione.

3.2 Procedimento logico

3.2.1 Specifiche

Il requisito di progetto più stringente è quello di non dover modificare quanto già in uso, se non per sostituire un passo della produzione con un altro più semplice ed immediato mantenendo gli stessi *input* ed *output*. **L'obiettivo è di conservare opportunamente ed automaticamente l'informazione temporale associata all'aggiornamento della diapositiva**. La decodifica degli istanti che interessano si può fare in tempo reale, oppure in fase di post-produzione qualora ci sia il modo di recuperare il valore del tempo trascorso dall'inizio della lezione o del singolo blocco. Considerando che il docente può commettere errori e che in tal caso viene a mancare la continuità temporale della registrazione, la prima ipotesi (tempo reale) è da scartare.

La registrazione BetaCam della lezione, oltre alle tracce audio e video separate, contiene una informazione temporale assoluta, con risoluzione di un trentesimo di secondo. Si tratta quindi di associare alla registrazione analogica BetaCam un segnale sincrono con l'aggiornamento della diapositiva, senza però intaccare le tracce audio e video originali. In sala regia dello studio di produzione è possibile miscelare segnali analogici in banda audio provenienti da svariate sorgenti: microfoni, riproduttori musicali, audio del computer di supporto al docente. La stessa operazione (miscelazione) è realizzabile anche per i segnali video. Se si considera poi la facilità con cui si possono generare in modo *custom* segnali audio rispetto a segnali video, la scelta della traccia audio per raggiungere l'obiettivo sembra la più vantaggiosa.

Si ha quindi a disposizione la traccia audio della registrazione BetaCam, a cui aggiungere in modo trasparente una informazione di servizio (segnalazione associata al canale trasmissivo) sincrona con l'istante di cambio diapositiva.

3.2.2 Esempi reali

Facendo una breve escursione nel mondo reale delle trasmissioni analogiche, si trovano alcuni esempi di informazioni digitali associate al canale trasmissivo.

3.2.2.1 Teletext

Quotidianamente si utilizza il servizio di *teletext*[12] fornito dalle emittenti televisive a diffusione circolare per mezzo di canali radio, le quali utilizzano la modulazione VSB per il video e FM per l'audio[13]. Per ragioni che esulano dal contesto di questo scritto, le trasmissioni televisive contengono un numero di righe superiore a quanto visualizzato dai normali televisori. Di conseguenza le informazioni testuali del *teletext* sono trasmesse utilizzando alcune delle 30 righe (su 625) non visualizzate da un televisore calibrato correttamente. Si trasmettono 4 pagine al secondo, per un totale di 40 caratteri per 24 righe [8]. Ipotizzando 8 bit per carattere si ottiene una velocità di:

$$8\ bit \times 40\ caratteri \times 24\ righe \times 4\ \ pagine/s = 30720\ bit/s$$

3.2.2.2 R.D.S.

L'equivalente del servizio *teletext* per le trasmissioni radiofoniche circolari in banda VHF (88-108 MHz) si chiama *R.D.S.* (Radio Data System). La velocità di trasmissione dei dati è nettamente inferiore rispetto al caso televisivo, ma adatta alle dimensioni limitate dello schermo del ricevitore (8 caratteri alfanumerici) e sufficiente per il tipo di informazioni inviate (frequenze alternative, nome programma, informazioni stradali [9]).

Il flusso di bit dell'RDS viene inserito su una sotto-portante a 57 kHz di un segnale radio FM, con una velocità di canale di 1187,5 bit al secondo. Togliendo la ridondanza aggiunta dai codici di correzione dell'errore, la velocità dati effettiva è di 730 bit/s [10].

3.2.2.3 C.T.C.S.S.

Limitando ad un solo bit la quantità di informazione da associare al canale si incontra un esempio di utilizzo ancora diffuso sebbene non largamente conosciuto: i ponti radio per trasmissioni analogiche.

Un ponte radio, nella sua versione più semplice (Figura 3.1), è costituito da un ricevitore e un trasmettitore opportunamente interconnessi e operanti su due frequenze diverse (di *uplink e* di *downlink*, separate da uno *shift*). Il ponte radio è posizionato in un luogo strategicamente conveniente, cioè più in alto rispetto agli utenti. Gli utenti

---

[12] Servizio conosciuto in Italia con il termine "televideo".

[13] VSB: Vestigial Side Band; una variazione della modulazione a banda laterale doppia (DSB) che limita a 5 MHz la larghezza di banda richiesta da una trasmissione video PAL. FM: modulazione di frequenza.

sono dotati di terminali portatili (ricetrasmettitori *half-duplex*) e comunicano tra di loro in modo asincrono.

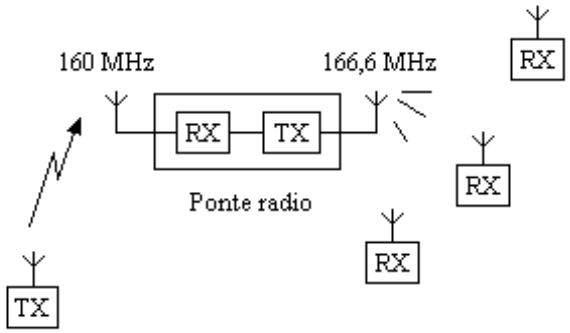

Figura 3.1 Schema di principio di un ponte radio

Data la posizione sopraelevata del ponte radio si rende necessario limitare il suo utilizzo solo agli utenti abilitati: in caso contrario qualunque segnale ricevuto dal ponte verrebbe ritrasmesso sulla frequenza di *downlink*; eventualmente bloccandone la funzionalità, creando interferenze o danneggiando l'amplificatore di potenza del trasmettitore. Negli anni in cui il numero di ponti radio per servizi privati e di pubblica utilità crebbe esponenzialmente (anni '70), la tecnologia non consentiva una estrema miniaturizzazione dei componenti elettronici.

Al fine di mantenere i terminali degli utenti leggeri, concettualmente semplici e con un consumo energetico limitato, la Motorola propose un sistema di controllo degli accessi basato su un tono a frequenza subsonica aggiunto all'unico canale informativo trasmesso: la voce [11]. Questo sistema è denominato CTCSS[14] ed ha portato all'adozione di una serie di toni subsonici standard ([12], [13]). Il segnale ricevuto dal ponte radio viene ritrasmesso solo se trasporta anche il tono prestabilito (Figura 3.2).

L'informazione associata al tono è relativa all'abilitazione del terminale utente ad utilizzare la risorsa condivisa, e si può rappresentare con un solo bit.

14 "Continuous Tone Controlled Signalling System"; sistema di segnalazione basato su un tono continuo.

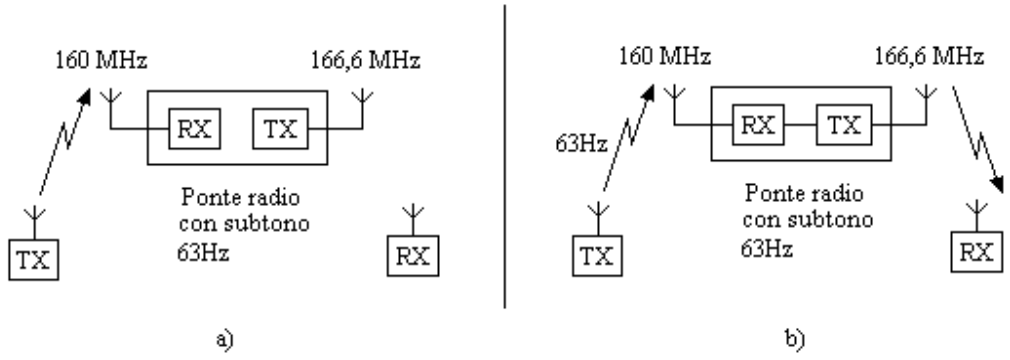

Figura 3.2 Utilizzo di un ponte radio protetto da tono subsonico

Infine, anche la segnalazione necessaria alla composizione del numero telefonico da chiamare da un telefono fisso è associata al canale, ma non in modo trasparente all'utente (si sentono infatti i "toni" DTMF). Allo stesso modo il suono che precede il segnale orario ufficiale (quello dell'Istituto Elettrotecnico Nazionale "Galileo Ferraris" di Torino) è una informazione binaria codificata, e associata al canale [14].

3.2.3 Psicoacustica

Per capire le motivazioni che hanno portato ad utilizzare toni a frequenza subsonica nei ponti radio (CTCSS) è necessario analizzare gli studi effettuati da Fletcher e Munson negli anni 50-60. Essi studiarono la risposta in frequenza dell'orecchio umano in condizioni di ascolto in campo libero, definendo sperimentalmente delle curve normalizzate a iso-intensità per toni puri [15].

Se si osserva il diagramma di Fletcher-Munson (Figura 3.3, [16]) si nota che l'orecchio umano ha una risposta in frequenza non piatta. In altri termini, ascoltando dei toni a frequenza diversa ma con lo stesso volume si ha la sensazione che essi abbiano intensità differenti. Prendendo come esempio un tono a 1000 Hz e uno a 100 Hz, un livello di intensità di 10 Phon per entrambi si raggiunge solo aumentando di 20 dB il tono più basso. Le stesse considerazioni valgono per le frequenze al di sopra dell'intervallo normalmente coperto dalla voce umana.

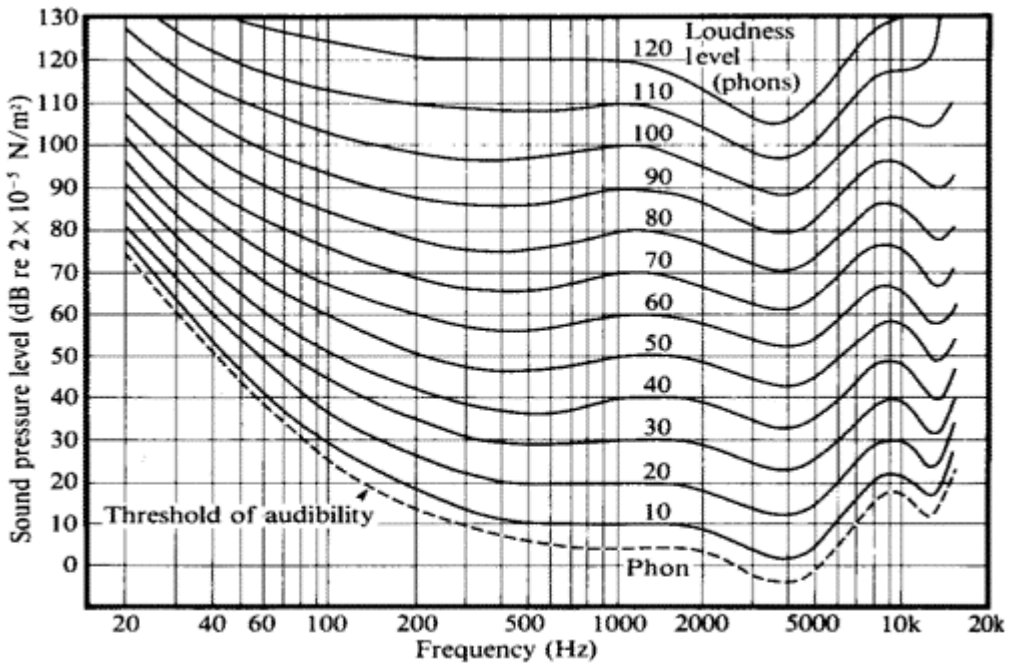

Figura 3.3 Diagramma di Fletcher-Munson dei livelli a iso-intensità in campo libero; rappresentazione grafica della sensibilità media dell'orecchio umano tra 20 Hz e 15 kHz.

Le caratteristiche dell'orecchio umano diagrammate da Fletcher e Munson, assieme ad altre proprietà uditive del medesimo organo studiate in seguito, sono attualmente alla base degli algoritmi di compressione audio di tipo MPEG ([17], [18], [19], [20]). In [21] si legge:

"L'algoritmo MPEG/audio comprime i segnali audio principalmente rimuovendone le componenti acusticamente irrilevanti. In pratica si avvantaggia dell'incapacità del sistema auditorio umano di rilevare il rumore di quantizzazione in condizioni di mascheramento sonoro. Il mascheramento è una proprietà percettiva del sistema auditorio umano, e si presenta ogni volta che un forte segnale audio rende impercettibili segnali più deboli temporalmente o spettralmente adiacenti. Una serie di esperimenti psicoacustici conferma questo fenomeno di mascheramento [22]."

"I risultati empirici mostrano che il sistema auditorio umano ha una risoluzione limitata e dipendente dalla frequenza. La dipendenza dalla frequenza può essere espressa in termini di larghezze di banda critiche che sono inferiori a 100 Hz per le più basse frequenze udibili e superiori a 4 kHz per le più alte. Il sistema acustico umano sfoca le componenti del segnale che cadono all'interno di una banda critica, sebbene la selettività in frequenza di questo sistema (compressione MPEG/audio, N.d.T.) sia molto migliore di una banda critica."

"Grazie alla potenza risolutiva del sistema auditorio umano, che è funzione dalla frequenza, la soglia di mascheramento del rumore a qualunque frequenza dipende solamente dall'energia del segnale in una banda di frequenza limitata alle vicinanze della frequenza stessa. […] MPEG/audio funziona suddividendo il segnale audio in sottobande di frequenza che approssimano le fasce critiche, e poi quantizzando ogni sottobanda in funzione dell'udibilità del rumore di quantizzazione all'interno di ogni fascia. Per una compressione più efficiente, ogni sottobanda deve essere quantizzata con un numero di livelli appena sufficiente a rendere non udibile il rumore di quantizzazione."

"Il modello psicoacustico analizza il segnale audio e calcola la quantità di mascheramento del rumore disponibile come funzione della frequenza ([23], [24], [25], [26]). La capacità di mascheramento di una determinata componente del segnale dipende dalla sua posizione in frequenza e intensità. Il codificatore usa queste informazioni per decidere come meglio rappresentare il segnale audio di ingresso con il numero limitato di bit di informazione disponibile."

3.2.4 Steganografia

Alla luce di quanto esposto in 3.2.3 qui sopra, si conclude che si può aggiungere ad una trasmissione o registrazione sonora dei toni a frequenza subsonica o ipersonica, associandovi a priori delle informazioni, a condizione che essi cadano nella banda passante del sistema totale, senza che siano percepiti da chi ascolta.

"Questa tecnica viene chiamata *steganografia* e consiste nell'inserire dei dati all'interno di un messaggio in un modo non percepibile da un osservatore esterno. In questo senso la *steganografia* differisce dalla crittografia in quanto il messaggio rimane inalterato e completamente significativo dal punto di vista percettivo [27]."

3.2.5 Larghezze di banda

La soluzione adottata per la protezione dei ponti radio (CTCSS) sfrutta contemporaneamente la limitata capacità uditiva dell'orecchio umano e le caratteristiche tecniche degli apparati ricetrasmittenti. Le applicazioni radiofoniche di qualità trasmettono (o registrano) un intervallo di frequenza compreso tra poche decine di Hz e un valore variabile tra 15 e 22 kHz in funzione della larghezza di banda massima ammessa per il canale trasmissivo utilizzato. La limitazione dell'intervallo sonoro avviene interponendo un filtro tra la sorgente audio e il modulatore, in trasmissione, e realizzando tale risposta in frequenza per l'amplificatore audio al ricevitore. I circuiti di trasmissione contengono ulteriori filtri, la cui banda passante è

più larga rispetto alla banda audio, specialmente verso le basse frequenze: si può ipotizzare che anche pochi Hz possano essere trasmessi con qualunque apparato commerciale su un qualunque mezzo trasmissivo.

3.2.6 Soluzione proposta

Considerando le conclusioni della *psicoacustica* (3.2.3 Psicoacustica), e facendo una analogia tra il ponte radio e il sistema di produzione di lezioni multimediali si giunge ad una soluzione per il problema della sincronizzazione di diapositive con flussi audiovisivi registrati. Si veda la Figura 3.4 per un parallelismo grafico tra i due sistemi.

3.2.6.1 Premesse

La sorgente audio primaria delle lezioni multimediali è sempre un microfono, anche se con una risposta in frequenza più ampia rispetto alla banda vocale (20 Hz – 20 kHz contro 300 – 3400 Hz). Il segnale audio, non filtrato, è sommato ad un segnale sinusoidale di ampiezza non trascurabile e quindi inviato al sistema di registrazione. Quest'ultimo, al pari dell'insieme trasmettitore-canale-ricevitore, ha una banda passante limitata sia da ragioni tecnologiche sia dal filtraggio dei segnali imposto dai costruttori.

La compressione dei flussi audio e video adottata nell'ultimo passo del processo produttivo è di tipo MPEG[15]. Come già visto, la caratteristica principale degli algoritmi di compressione audio (e video) MPEG è quella di non codificare le componenti spettrali del segnale[16] con ampiezza tale da essere irrilevante per l'orecchio umano, secondo un modello *psicoacustico* basato anche sulle conclusioni di Fletcher e Munson [28]. Si realizza così una banda passante variabile dinamicamente in funzione delle ampiezze delle singole componenti spettrali poste agli estremi dell'intervallo di udibilità.

---

[15] Moving Picture Expert Group è un gruppo di lavoro che sviluppa standard per la compressione di audio e video numerici. Opera all'interno dell'Organizzazione per gli Standard Internazionali (OSI o ISO)

[16] Un segnale può essere rappresentato nel dominio della frequenza tramite la trasformazione di Fourier (vedi oltre). Si ottiene così lo spettro di frequenza del segnale.

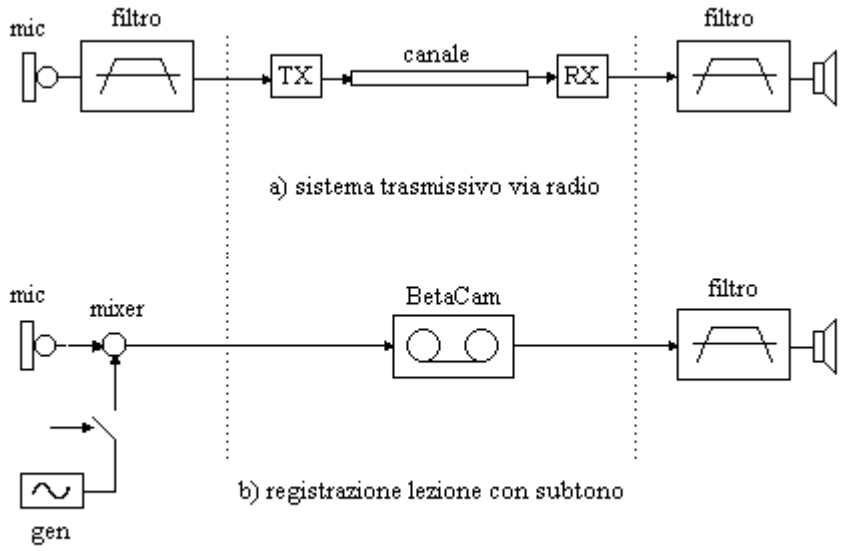

Figura 3.4 Parallelismo tra sistema trasmissivo e sistema di produzione di lezioni multimediali con sincronia a tono subsonico

Il sistema di riproduzione delle lezioni multimediali raramente è di qualità tale da riprodurre fedelmente quanto registrato in quanto costituito da cuffie [29] o altoparlanti di dimensioni ridotte (il filtro di uscita rappresentato in figura comprende anche lo scarso rendimento dei diffusori alle basse/alte frequenze). Combinando quindi le risposte in frequenza tipiche dell'orecchio umano e dei diffusori sonori su cui viene ascoltato il prodotto finito, si ha una attenuazione intrinseca del sistema per quelle frequenze che non rientrano nella banda vocale.

3.2.6.2 Conclusioni

Dai manuali dei videoregistratori SONY BetaCam [30] si apprende che la banda passante audio è compresa tra 50 Hz e 15 kHz (+2 dB, -3 dB), cioè essa si estende molto al di sotto della soglia di udibilità per intensità confrontabili con quelle di un segnale audio. Anche la banda passante delle comuni schede audio per personal computer scende fino a 20 Hz circa (sia per la riproduzione sia per la registrazione, [31], [32]).

Si prospetta questa soluzione:

- **generare un tono sinusoidale a frequenza subsonica in concomitanza del cambio di diapositiva**
- **registrarlo sulla cassetta BetaCam**

- **recuperare la sincronia in un secondo tempo, in fase di post-produzione**

La scelta di un tono al di sotto dell'intervallo di frequenze in cui cade la voce umana, e non al di sopra, è dovuta a più fattori contemporaneamente.

- L'intera lezione viene convertita in formato numerico per la distribuzione al pubblico: bisogna quindi rispettare la *frequenza di Nyquist*[17] al fine di evitare il fenomeno dell'*aliasing*[18] [33].
- La desensibilizzazione dell'orecchio umano è più marcata alle basse frequenze
- I diffusori utilizzati per la riproduzione delle lezioni saranno di dimensioni limitate, quindi più propensi a non filtrare intrinsecamente i toni alti (rendimento maggiore alle alte frequenze per bassi volumi)

Utilizzando una tecnica *steganografica* (cfr. 3.2.4) si propone quindi di associare il segnale di sincronia al normale audio della lezione tramite l'utilizzo di un tono puro (sinusoidale) a frequenza subsonica. Il tono ha frequenza e durata prestabilite, ampiezza costante a regime. In fase di post-produzione si cerca nella traccia audio la presenza dei toni e si calcola la loro distanza temporale dall'inizio della lezione (che è un valore noto). Un file contenente i tempi ottenuti, formattati opportunamente, costituisce l'uscita del processo di riconoscimento e l'ingresso del passo successivo di post-produzione (sincronizzazione audio/video con diapositive e generazione dell'ASF).

Infine la codifica MPEG attuata sulla traccia audio, sulla base di considerazioni *psicoacustiche*, elimina il tono a frequenza subsonica in quanto sotto la soglia di udibilità e considerato pertanto alla stregua del rumore: l'utente finale non può eventualmente percepire il segnale di sincronia perché assente sul prodotto finito.

---

[17] Secondo il *teorema di Nyquist* (o *teorema del campionamento*) un segnale analogico è completamente determinato dai suoi campioni se la frequenza di campionamento è pari ad almeno il doppio della massima frequenza in esso contenuto. La *frequenza (critica) di Nyquist* è pari alla metà della frequenza di campionamento, e la banda del segnale deve essere limitata a tale valore.

[18] *Aliasing*: fenonemo (indesiderato) di riporto delle componenti spettrali che giacciono oltre la frequenza critica di Nyquist f_c all'interno della banda analizzata (da $-f_c$ a $+f_c$). Nel dominio temporale si manifesta con un riporto delle code del segnale troncato.

Dispositivo	Limite inferiore (Hz)	Limite superiore (Hz)
Microfono [34]	20	16000
Mixer (-1 dB)	20	20000
Videoregistratore BetaCam [30]	50	15000
Scheda acquisizione (-0.6 dB) [35]	20	20000
Scheda audio (registrazione)	20	20000
Scheda audio (riproduzione)	20	20000
Codifica audio MPEG	variabile	variabile
Cuffie	70	22000
Cavi	*0*	*F.ne della lunghezza*
Orecchio umano	*200*	*14000*

Tabella 3: tabella riassuntiva delle bande passanti audio (-3 dB se non specificato altrimenti) dei dispositivi coinvolti nella produzione e utilizzo delle lezioni multimediali. Le ultime due righe riportano componenti del sistema totale che non si possono definire *dispositivi*, le cui bande passanti vanno comunque tenute in considerazione.

3.3 Generazione segnale di sincronia

Durante la registrazione di una lezione è necessario generare il segnale di sincronia ad ogni cambio di diapositiva. Attualmente la maggior parte delle presentazioni che utilizzano il supporto del computer vengono realizzate con Microsoft *PowerPoint*. Tale software permette di associare suoni ad ogni evento della presentazione, dalla visualizzazione di un singolo carattere al cambio diapositiva. I suoni possono essere scelti da una libreria o da fonti esterne, purché siano in un

formato riproducibile da *PowerPoint*, quindi *WAVE*[19].

In *PowerPoint* l'inserimento di un effetto sonoro uguale per tutte le diapositive di una presentazione richiede pochissime operazioni (Figura 3.5 e Figura 3.6).

La semplicità imposta alla grafica di una lezione multimediale non prevede l'utilizzo di effetti sonori particolari e tantomeno diapositive con animazioni: l'unico soggetto in movimento videoregistrato è la figura del docente. E' quindi possibile affidare al visualizzatore (*PowerPoint* stesso) il compito di generare il segnale di sincronismo.

Qualora il software della presentazione non permetta di associare un effetto sonoro all'azione di cambio diapositiva, è necessario realizzare un software che generi il tono di sincronia sotto certe condizioni (pressione di un tasto, click sul mouse, …).

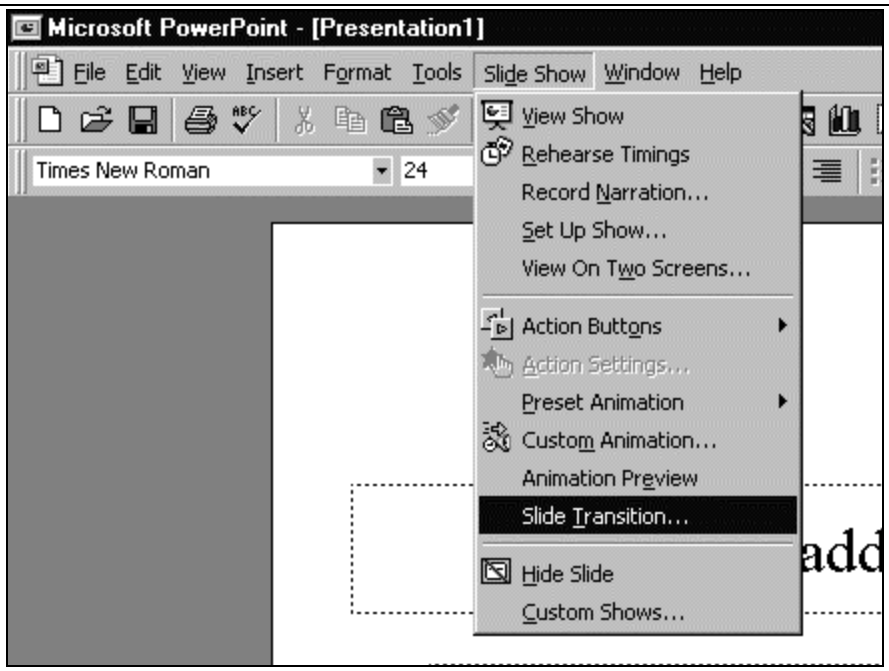

Figura 3.5 Primo passo per aggiungere una tantum un effetto sonoro valido per tutte le diapositive di una presentazione *PowerPoint*

19 *WAVE* è un formato di file creato da Microsoft che è divenuto uno standard come mezzo di scambio di contenuti sonori tra piattaforme software diverse. Oltre ai dati audio non compressi, un file *WAVE* contiene informazioni sul numero di tracce (modo o stereo), frequenza di campionamento e risoluzione in bit per campione.

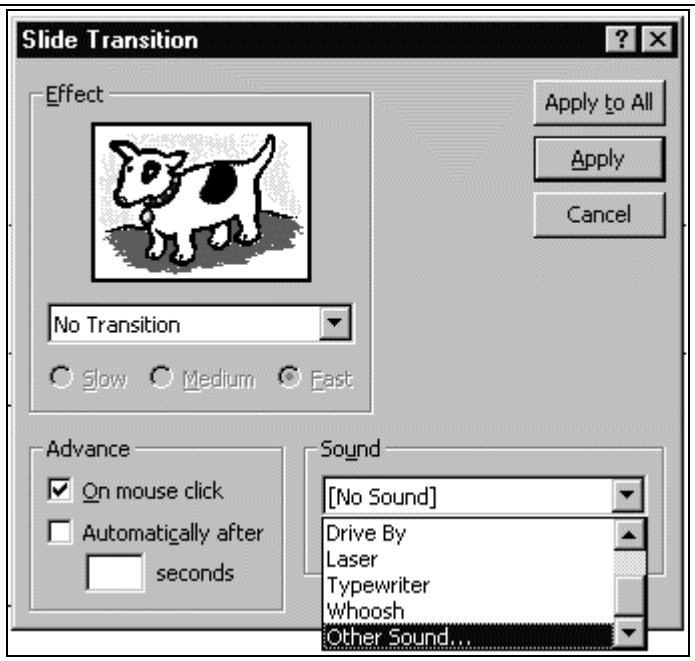

Figura 3.6 Selezione del file sonoro da associare ad ogni diapositiva di una presentazione *PowerPoint*; secondo passo.

La frequenza e la durata del tono da generare sono stati definiti per via sperimentale utilizzando le attrezzature professionali messe a disposizione dal Ce.Te.M. Per la scelta di un intervallo di frequenze valide si è dovuto tenere conto:

- del limite inferiore della banda passante per le registrazioni BetaCam e la riproduzione delle schede sonore per PC, al fine di definire un valore minino dell'intervallo
- dei livelli di udibilità dell'orecchio umano, per la determinazione del limite superiore
- degli eventuali disturbi intrinseci del sistema, che avrebbero potuto creare *battimenti* in frequenza (ronzii a 50 Hz o multipli, rumore introdotto dal bus del computer, rumorosità dei componenti analogici, …)

Non è invece possibile determinare a priori quale sarà il tipo di filtraggio introdotto dalla compressione MPEG, e scegliere di conseguenza la frequenza e l'ampiezza del tono. In ogni caso quest'ultimo parametro influisce solo sull'eventuale rimozione del segnale di sincronia, ormai inutile, dal prodotto finito.

La durata minima del tono di sincronismo non è critica, sebbene debba essere confrontabile con la lunghezza temporale dei campioni analizzati in una singola trasformata di Fourier (cfr. 3.4 Riconoscimento segnale di sincronia). Al fine di minimizzare la possibilità di errori di decodifica si sceglie una durata equivalente ad

un numero di campioni (almeno) doppio rispetto alla lunghezza della singola DTFT (cfr. 3.4.1):

$$Durata \geq 2 \times \frac{8192\ bit}{44100\ bit/s} \cong 0.375\ s$$

Il risultato di 0.375 secondi vale per una lunghezza della DTFT di 8192 bit, campionati a 44.1 kHz. La durata massima è imposta invece dalla frequenza con cui si susseguono le diapositive, che solitamente sono esposte per 30 o più secondi. L'intervallo tra due toni di sincronia deve essere almeno pari alla durata minima calcolata come sopra.

L'ampiezza del tono all'uscita del miscelatore audio può variare tra il minimo indispensabile ad una decodifica corretta e il valore che non produce distorsione per saturazione dell'audio utile (da parte di una delle componenti del ciclo produttivo; quali videoregistratore, scheda di acquisizione, terminale utente di riproduzione). L'intensità dell'audio utile di una lezione è funzione dei livelli impostati sul miscelatore audio in sala regia durante la registrazione (livello per singolo ingresso, volume dell'uscita miscelata). Allo stesso tempo il volume di riproduzione della scheda sonora dell'elaboratore su cui sono visualizzate le diapositive varia per ogni modello: non è quindi possibile effettuare misure di ampiezza assolute per il tono subsonico di sincronia. Spetta invece agli operatori di regia scegliere i livelli di miscelazione dei due o più segnali, fermo restando che esiste un ampio margine di movimento.

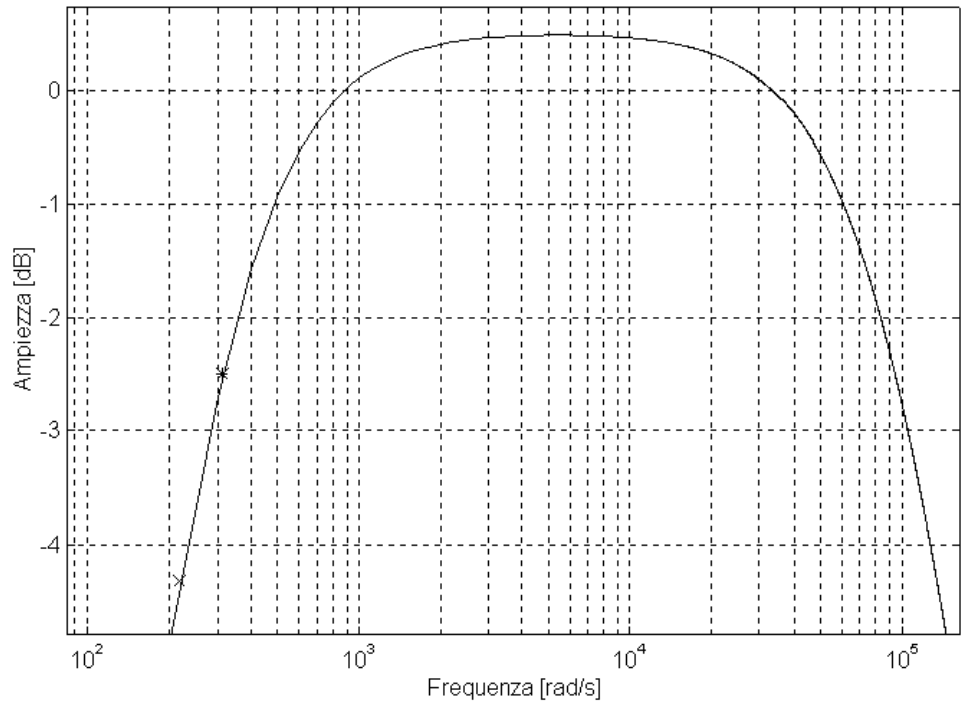

Figura 3.7 Andamento simulato della banda passante audio del registratore BetaCam. I punti evidenziati rappresentano, da sinistra, 35 Hz e 50 Hz.

Durante i test effettuati è stato scelto un tono sinusoidale con le seguenti caratteristiche:

- Frequenza: 35 Hz
- Durata: 2 secondi
- Formato: PCM a 16 bit per campione (WAVE), campionamento a 44100 Hz, monoaurale
- Ampiezza massima: 50% di quanto consentito dal formato WAVE
- Transitorio: variazione da/per ampiezza nulla all'inizio/fine del tono per eliminare degli impulsi udibili nella riproduzione del prodotto finito

La scelta di una frequenza al di sotto della banda passante dichiarata per il videoregistratore BetaCam è motivata dal fatto che, se a 50 Hz l'attenuazione è di 3 dB, a 35 Hz ci sono solo 1.8 dB aggiuntivi (Figura 3.7 [20]). Una attenuazione totale di

[20] Ipotizzando una risposta in frequenza del sistema videoregistratore/cassetta BetaCam con uno zero nell'origine, un polo a 50 Hz e uno a 15 kHz. Valori simulati con MATLAB.

4.8 dB rispetto alla media della banda passante è ancora accettabile ed eventualmente compensabile incrementando il volume del tono in fase di registrazione.

Inoltre 35 Hz non sono prossimi della frequenza di rete (50 Hz), che potrebbe creare interferenze e segnali di sincronia inesistenti. La durata di due secondi è ampiamente oltre il limite minimo calcolato sopra, ed è adatta ad un ampio raggio di coppie "lunghezza FFT/frequenza di campionamento".

Il file contenente il tono è stato generato con il software *freeware NCH Tone Generator* [36] (Figura 3.8). Con Adobe Premiere si è provveduto a modificare il file inserendo l'evanescenza all'inizio e alla fine del tono.

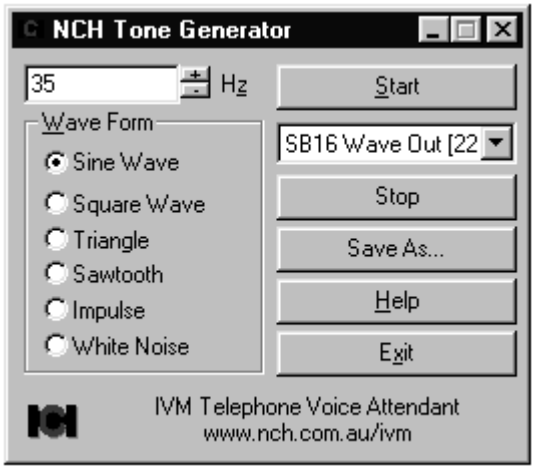

Figura 3.8 Software utilizzato per la generazione del file contenente il tono a frequenza subsonica

Il file generato deve avere le stesse informazioni per entrambi i canali audio in modo da produrre un segnale monoaurale: qualora la registrazione audio in BetaCam avvenisse su un solo canale (sinistro o destro), il segnale di sincronismo per le diapositive non verrebbe perso.

3.4 Riconoscimento segnale di sincronia

3.4.1 Analisi spettrale

Il problema del riconoscimento della presenza di una determinata frequenza all'interno di un segnale analogico ha alcune soluzioni. Si escluda il caso in cui il segnale di origine contiene solo la componente sinusoidale di interesse (situazione teorica che non trova mai riscontro nel mondo reale), mentre si considerino segnali complessi a banda limitata con estensione temporale non trascurabile ma non infinita.

3.4.1.1 Analisi hardware

Volendo affidare il riconoscimento ad una soluzione puramente hardware, il vasto mercato dei semiconduttori propone dei decodificatori di tono con ingresso analogico ed uscita binaria (livello logico "1" in presenza di tono, "0" altrimenti). Il loro funzionamento si basa su un caso particolare di utilizzo di un PLL[21], e la calibrazione della frequenza desiderata e della sensibilità avviene con delle regolazioni manuali su componenti elettronici (potenziometri o condensatori variabili).

La Figura 3.9 riporta lo schema elettrico di un decodificatore di tono nell'intervallo delle frequenze audio basato sull'integrato NE567 della Philips [38]. Questa soluzione, oltre a non presentare la flessibilità necessaria e una larghezza di banda sufficientemente stretta, richiede comunque lo sviluppo di un software per la gestione dell'uscita binaria del discriminatore di frequenza e l'interfacciamento con il generatore di riferimento temporale.

Si è preferita una soluzione interamente software, che sfruttasse appieno le capacità di calcolo offerte dagli elaboratori di ultima generazione.

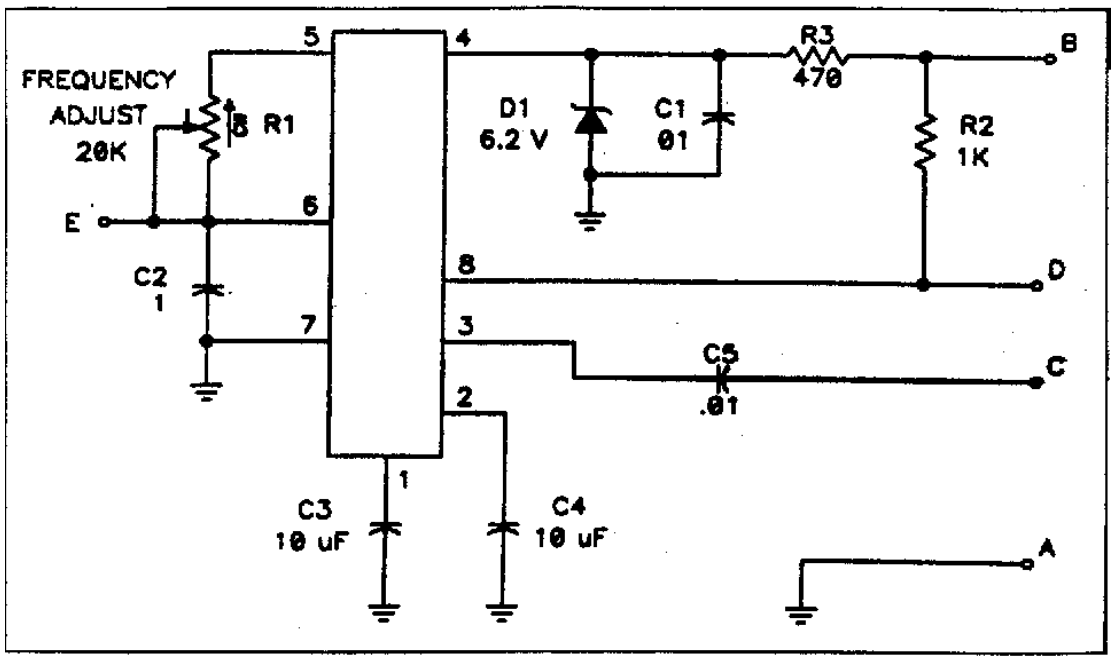

Figura 3.9 Esempio di decodificatore di tono con NE567 della Philips [37]

[21] Phased Locked Loop, anello ad aggancio di fase. Particolare configurazione ad anello di un comparatore di fase, un oscillatore di precisione e un divisore di frequenza utilizzata per stabilizzare nel tempo oscillatori liberi.

3.4.1.2 Analisi software

Matematicamente, il riconoscimento della presenza di uno o più toni all'interno di un segnale complesso si effettua valutando l'ampiezza della componente spettrale ad essi corrispondente [39]. In altre parole effettuando un'analisi spettrale del segnale analogico di partenza tramite una trasformazione di Fourier. "L'approccio offerto da questo strumento matematico (l'analisi di Fourier, N.d.T.) può essere applicato, nelle sue varie forme, a numerosi tipi di segnale, inclusi quelli a tempo continuo o discreto, periodici o aperiodici ed anche a due dimensioni[40]."

Per analizzare la copia numerica di un segnale aperiodico quantizzato, si utilizza la versione della trasformata di Fourier per i segnali a tempo discreto: Discrete Time Fourier Transform (DTFT) [41]. Si assuma che l'insieme di dati numerici rappresenti un segnale

1. a banda limitata (rispetta il teorema del campionamento)
2. campionato ogni Δ secondi
3. reale (i campioni non sono valori complessi tipo *Re + j Im*)

Se la prima condizione è verificata non si ha il fenomeno dell'*aliasing*, cioè il riporto delle componenti spettrali oltre la frequenza critica di Nyquist f_c all'interno della banda analizzata (da $-f_c$ a $+f_c$).

Per la seconda condizione, la stima spettrale che si ottiene dalla DTFT è valida solo per i valori delle frequenze discrete:

$$F_n = \frac{n}{N \times \Delta} \qquad n = -\frac{N}{2},...,\frac{N}{2}$$

Dove N è il numero di campioni su cui si effettua la DTFT [42], o lunghezza della trasformazione.

La terza assunzione, verificata per ogni campionamento di segnali del mondo reale, rende più efficiente l'algoritmo di trasformazione, il quale sfrutta inoltre la proprietà di simmetria:

$$\left(F_{N-n}\right)^* = F_n \qquad n = 0,...,\frac{N}{2}$$

In questo modo è necessario conservare solo la metà positiva dell'intero spettro, data la simmetria rispetto all'origine 0 + j0: si risparmia memoria e cicli di CPU (preziosi per le analisi in tempo quasi-reale).

3.4.1.3 Windowing

Esiste però una limitazione a sfavore di questa tecnica di analisi. Per ragioni fisiche e semplicità computazionale si analizza un insieme ristretto dei dati campionati (N campioni). Ciò equivale a moltiplicare, nel dominio del tempo, la traccia di ingresso che teoricamente si estende da $t=-\infty$ a $t=+\infty$ con una finestra rettangolare ampia tanto quanto i campioni considerati ad ogni analisi spettrale ($t_w = N*\Delta$), come in Figura 3.10.

Per le proprietà delle trasformazioni di Fourier, una moltiplicazione nel tempo diviene una convoluzione in frequenza. Lo spettro di una finestra rettangolare è una funzione di tipo *sinc*: la stima spettrale ottenuta alle singole frequenze discrete per un segnale troncato nel tempo non è accurata, in quanto ogni componente dello spettro contiene delle "perdite" provenienti da numerose componenti adiacenti ([43], [44]).

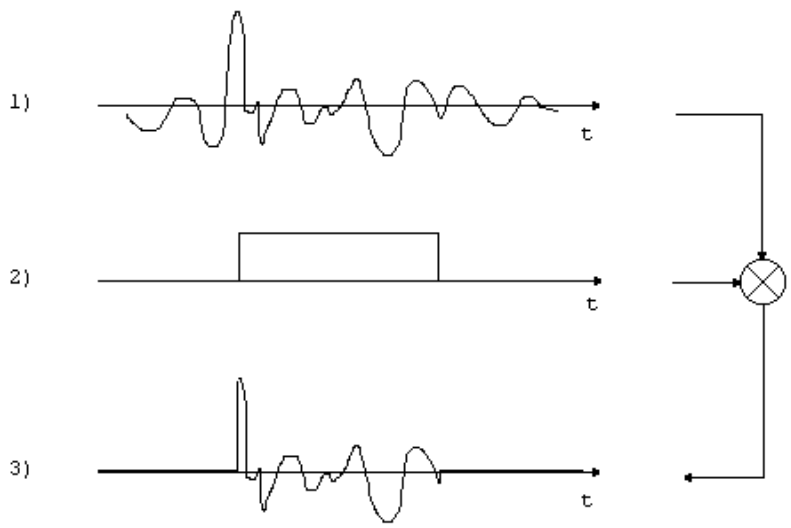

Figura 3.10 Moltiplicazione del segnale analogico 1) per una finestra rettangolare 2) larga $t_w = N*\Delta$ secondi. Il risultato 3), non nullo solo nell'intervallo t_w, viene suddiviso in N campioni, distanti $\Delta = (f_{campionamento})^{-1}$ secondi l'uno dall'altro (non rappresentato).

Si rende necessario modificare artificialmente la forma della finestra utilizzata al fine di minimizzare le discontinuità del segnale nel dominio del tempo. Non potendo agire direttamente sulla finestra, si provvede a moltiplicare i campioni per una ulteriore finestra, di pari durata, ma di forma più consona allo scopo [45].

Data la non criticità delle misure spettrali da effettuare per il riconoscimento del tono di sincronia, si è scelta la finestra triangolare o di Bartlett [46]: l'allargamento della riga spettrale diviene trascurabile già a distanza superiore a 2 frequenze discrete [47]. Allo stesso tempo l'allargamento è valido per tutte le componenti spettrali reali (quelle che risulterebbero facendo l'analisi spettrale del segnale non campionato, ottenendo quindi uno spettro di frequenze continuo). Questo effetto di allargamento incrementa involontariamente il numero di componenti spettrali rappresentate all'interno di un intervallo di frequenza discreta. Assumendo inoltre che non vi siano componenti spettrali di ampiezza confrontabile con quella del tono subsonico utilizzato nelle sue immediate vicinanze (cfr. § 5.2), questa proprietà della finestra di Bartlett permette di riconoscere il tono anche se non rispetta la relazione 3.2.

3.4.1.4 Procedure software

L'algoritmo software di trasformazione di Fourier scelto si compone di due funzioni in linguaggio C: `fft(...)` per il calcolo di una generica trasformazione di Fourier di tipo FFT[22] per campioni complessi [48], richiamata da una procedura per formattare opportunamente i dati reali (`realft(...)`) in modo da ottenere un risultato coerente nel dominio delle frequenze [49].

L'algoritmo FFT utilizzato si basa sulla proprietà della trasformata discreta di Fourier, lunga N, di poter essere scomposta nella somma di due trasformate lunghe N/2 in modo ricorsivo, fino a trasformate di lunghezza unitaria [50]. La complessità computazionale di un algoritmo di trasformata veloce (FFT) è dell'ordine di $N*\log_2(N)$ contro N^2 dell'algoritmo generico a condizione che N, lunghezza della trasformata, sia una potenza intera di 2.

Il formato dei dati richiesto dalle due funzioni di trasformazione è `float`, e in assenza di segnale i campioni hanno valore nullo (no *zero offset*; niente componente continua sul segnale).

3.4.2 Algoritmo di ricerca

L'algoritmo di ricerca e riconoscimento del tono di sincronia individua le variazioni di ampiezza della componente spettrale alla frequenza (subsonica) prescelta. In corrispondenza di una variazione si verifica inoltre che il tono sia

[22] Fast Fourier Transform, trasformata veloce di Fourier. In questo documento si fa riferimento indifferentemente alla DTFT o alla FFT, indicando la stessa trasformazione.

presente per un certo numero di campioni consecutivi, multiplo intero della lunghezza del buffer di ingresso.

Si definisce così per il software una durata minima del tono, che deve essere inferiore alla durata effettiva dello stesso. Infatti, come si è visto nel paragrafo 3.3, il tono di sincronia inserito sulla traccia audio della registrazione presenta una evanescenza di ingresso e di uscita (*fade in* e *fade out*). Tale evanescenza elimina dei fastidiosi impulsi udibili sul prodotto finale, ma riduce la durata utile del tono (ampiezza oltre il livello di soglia).

Al momento di analizzare un blocco di dati, può succedere che:

1. l'istante di inizio del tono coincida con il primo campione dei dati da analizzare (con probabilità pari all'inverso della lunghezza N della FFT)
2. l'istante di inizio del tono non coincida con il primo campione dei dati da analizzare

Nel secondo caso l'algoritmo potrebbe non rilevare la presenza del tono (densità spettrale media sulla durata del blocco inferiore alla soglia).

Quando l'applicativo verifica la presenza del tono per il tempo minimo prestabilito, provvede a calcolare il tempo di inizio del tono relativo all'inizio del blocco sotto esame secondo la relazione

```
(tempo inizio tono) = (numero prossimo campione) *
(intervallo campionamento) - (offset iniziale) -
(lunghezza software tono)
```

dove:

- `numero prossimo campione` è il numero assoluto che identifica il prossimo campione che verrà letto dal file, cioè la posizione del puntatore al flusso di ingresso
- `intervallo campionamento` è la spaziatura temporale tra i campioni
- `offset iniziale` è il tempo di inizio del flusso se non nullo (di solito è nullo)
- `lunghezza software tono` è il tempo che l'applicativo impiega prima di decidere la validità del tono

Per aumentare l'affidabilità del processo di riconoscimento, la durata reale del tono di sincronia deve essere superiore a quella imposta al software per la validazione di un segnale. Il software deve altresì attendere che il livello della componente spettrale controllata scenda sotto la soglia, prima di validare un ulteriore tono di sincronia. In questo modo se si cerca via software un tono di un secondo (44100/4096

= 10.8 ⇒ 11 analisi spettrali con campionamento a 44.1 kHz e buffer da 4kbyte) e il tono reale è di 10 secondi, l'algoritmo individua, correttamente, solo un segnale di sincronia. Il diagramma di flusso per l'algoritmo di riconoscimento del tono è rappresentato in Figura 3.11.

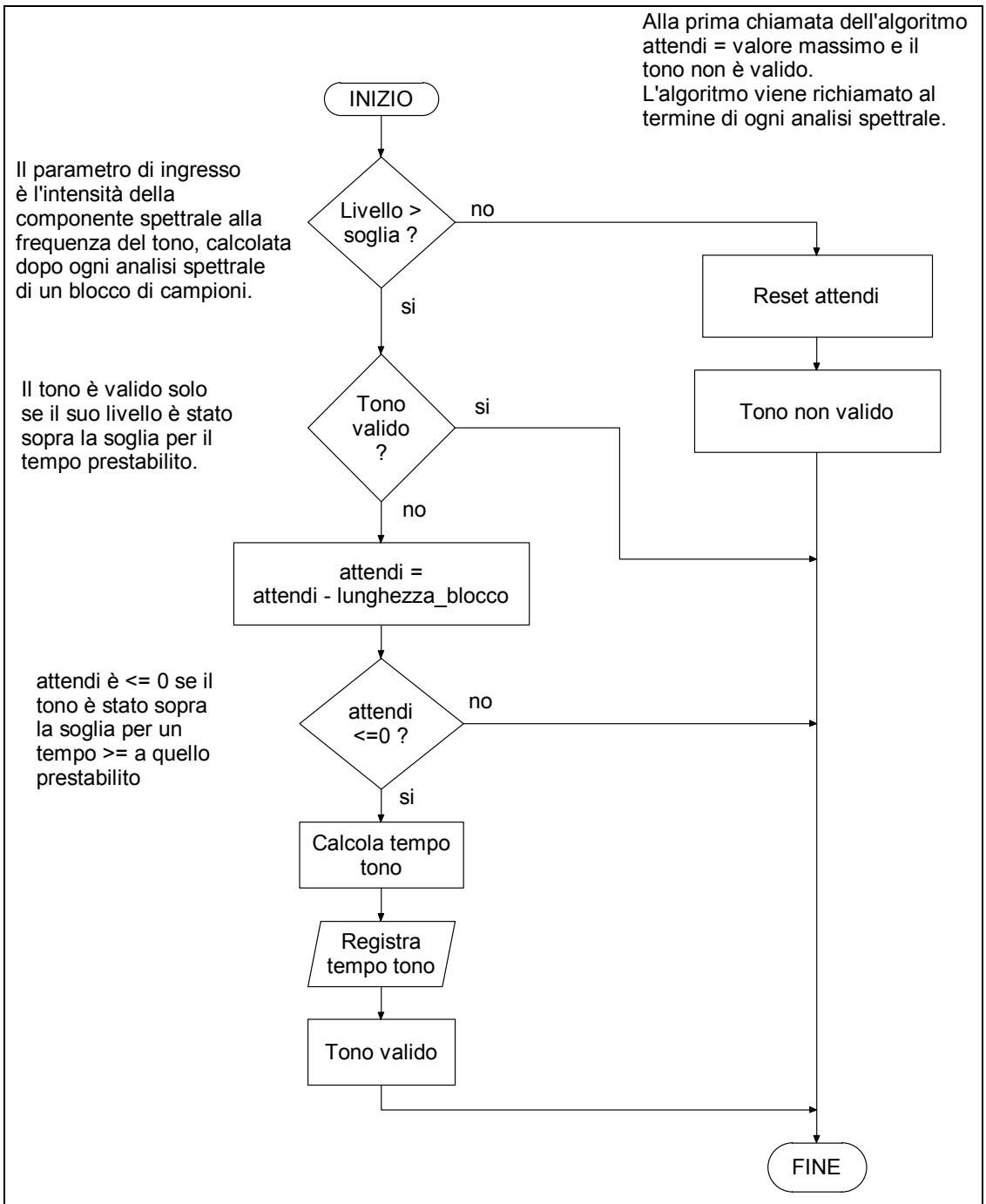
Alla prima chiamata dell'algoritmo attendi = valore massimo e il tono non è valido.
L'algoritmo viene richiamato al termine di ogni analisi spettrale.
INIZIO
Il parametro di ingresso è l'intensità della componente spettrale alla frequenza del tono, calcolata dopo ogni analisi spettrale di un blocco di campioni.
Livello > soglia ?
no
si
Reset attendi
Il tono è valido solo se il suo livello è stato sopra la soglia per il tempo prestabilito.
Tono valido ?
si
Tono non valido
no
attendi = attendi - lunghezza_blocco
attendi è <= 0 se il tono è stato sopra la soglia per un tempo >= a quello prestabilito
attendi <=0 ?
no
si
Calcola tempo tono
Registra tempo tono
Tono valido
FINE

Figura 3.11 Diagramma di flusso dell'algoritmo per determinare la validità del segnale di sincronia

4 Applicativi software

Il processo di implementazione della soluzione proposta, cioè di scrittura di un software preposto al riconoscimento del tono nascosto nella traccia audio, ha portato alla realizzazione di due applicativi. Mentre il codice per il riconoscimento del tono è uguale nei due applicativi (trasformata veloce di Fourier, cfr. 3.4), diverso è il metodo di interfacciamento con il sistema produttivo e diverso è il sistema operativo supportato.

Il primo applicativo, in ordine temporale, viene denominato "generico" perché utilizza librerie standard del linguaggio C: può funzionare su svariate piattaforme hardware e software con minime modifiche delle funzioni di I/O.

Il secondo, al contrario, viene denominato "specifico" perché è costruito attorno al processo produttivo descritto nel Capitolo 2 ed utilizza librerie disponibili solo per Windows.

4.1 Applicativo software generico

Come primo passo dell'implementazione software della soluzione proposta si è voluto minimizzare alcune difficoltà di programmazione, al fine di arrivare più rapidamente ad un applicativo funzionante che verificasse la validità di tutte le ipotesi fatte.

Il tono di sincronia è immagazzinato nella traccia audio della registrazione BetaCam. E' necessario interfacciare il videolettore BetaCam con il software preposto al recupero della sincronia. L'interfacciamento comprende sia la parte di comunicazione seriale sia l'uscita audio del lettore BetaCam.

Per il riconoscimento del tono di sincronia si analizza (tramite software) in tempo reale il contenuto spettrale della traccia audio riprodotta, la quale è riportata all'ingresso analogico di una scheda audio per PC (Figura 4.1). Si è visto infatti che la banda passante per tali dispositivi scende fino a circa 20 Hz, rendendoli quindi adatti a questo tipo di applicazione.

Si automatizza così quanto già effettuato dall'operatore in fase di post-produzione, rendendo meno aleatori i risultati e risparmiando al centro di produzione delle preziose ore-uomo.

I sistemi operativi non consentono a più applicazioni software di utilizzare contemporaneamente gli stessi dispositivi hardware: di conseguenza la soluzione

proposta non può operare parallelamente alla fase di acquisizione, ma deve necessariamente ripetere la riproduzione da cassetta della lezione.

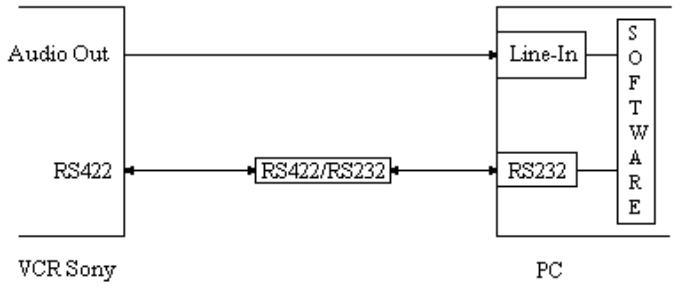

Figura 4.1 Esemplificazione delle connessioni tra videoregistratore BetaCam e PC

La documentazione del protocollo di comunicazione seriale SONY è disponibile su Internet [51] ed è riportata in Appendice A. Mediante comunicazione con il lettore BetaCam è possibile controllare tutti i parametri per la riproduzione del nastro, così come recuperare l'informazione temporale associata al singolo quadro video (*timecode*). Associando quest'ultima informazione alla presenza del tono sulla traccia audio si calcola il valore dell'istante di cambio diapositiva relativo all'inizio della lezione (o blocco), e si genera il file contenente i tempi da passare al successivo passo produttivo.

Particolari complicazioni implementative si riscontrano solitamente nella gestione dei dispositivi di interfaccia di un personal computer. Al fine di minimizzare i tempi da dedicare alla ricerca di malfunzionamenti software per la parte di comunicazione, si è scelto di sviluppare l'applicativo dimostrativo in Linux. Il linguaggio di programmazione scelto è il C, e si è mantenuta la massima compatibilità con la versione ANSI. Data la natura Open Source di Linux, esistono svariate fonti di documentazione relative alla programmazione di qualunque periferica supportata, così come esempi applicativi. Per Linux ogni dispositivo interno od esterno è accessibile utilizzando le stesse funzioni della gestione dei *file* (`open`, `close`, `read`, `write`). (Riferimenti sulla comunicazione seriale in Linux: [52], [53], [54].) Se la periferica è supportata dal *kernel* del sistema operativo, spetta al *driver* inserito nel *kernel* di convertire le chiamate a funzioni di interfaccia standard in istruzioni valide per lo

specifico dispositivo: il software prodotto non ha così bisogno di modifiche qualora si aggiornassero i dispositivi o la versione di Linux (distribuzione e/o versione del kernel) [55].

Il software consiste in due parti distinte preposte a funzioni diverse, ma interagenti: il processo per la comunicazione seriale con il videoregistratore e il processo per la ricerca del tono (DTFT).

La comunicazione seriale con il videoregistratore richiede l'implementazione di un insieme minimo delle istruzioni del protocollo Sony. Sono infatti sufficienti i comandi per avvolgere il nastro ad un valore predefinito di *timecode* ("Cue up with data"), avviare ("SyncPlay") e fermare ("Stop") la riproduzione, richiedere il valore attuale di timecode ("Current Time Sense") e verificare lo stato del dispositivo remoto ("Status Data"). Con un totale di 5 comandi, gestendo le relative risposte ed eccezioni fornite dal videoregistratore, si percorre tutta la registrazione alla ricerca del tono di sincronia. In Figura 4.2 è riportata la sola sequenza dei comandi da impartire al videoregistratore per riprodurre una lezione composta da un blocco unico.

4.1.1 Temporizzazioni

Il riconoscimento del tono si effettua tramite la procedura di trasformata veloce di Fourier descritta in "3.4 Riconoscimento segnale di sincronia". Il processo preposto alla DTFT attende il riempimento del *buffer* della scheda audio [56] per poi manipolare opportunamente il segnale audio campionato e verificare l'intensità della componente spettrale contenente la frequenza del tono subsonico. Allorché l'intensità rimane superiore al livello di soglia (definito dall'operatore) per un determinato numero di istanti di campionamento, si legge il *timecode* attuale dal videoriproduttore e si aggiorna il file di *output*.

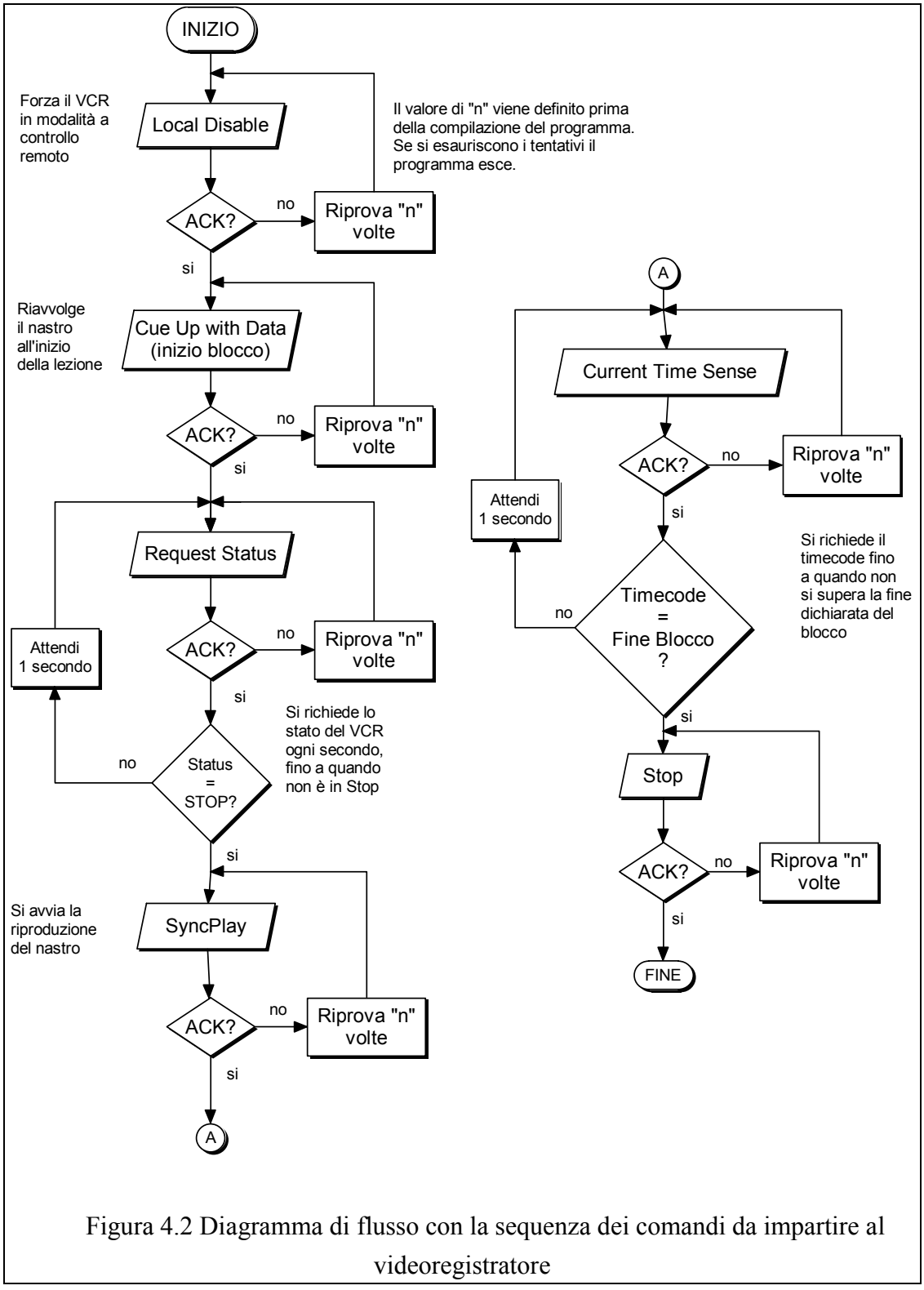

Figura 4.2 Diagramma di flusso con la sequenza dei comandi da impartire al videoregistratore

Dovendo lavorare su un flusso di dati in tempo reale e allo stesso tempo interagire con un dispositivo remoto, il software dimostrativo è composto da due processi[23] dedicati alle due operazioni: "DTFT" e "comunicazione". I requisiti temporali dell'applicativo sono i seguenti.

DTFT

Si ipotizzi una frequenza di campionamento di 44100 Hz e un buffer[24] di 4096 byte. La DTFT si effettua quindi su 4096 byte. Ciò porta ad una risoluzione nel dominio delle frequenze di 44100 [bit/s] / 4096 [bit] ≅ 10.76 Hz. Analogamente, il riempimento del buffer richiede

$$\frac{4096\ bit}{44100\ bit/s} \cong 92.8\ ms \qquad (4.1)$$

Questo è il tempo massimo a disposizione per elaborare i campioni e calcolarne la densità spettrale senza incorrere nella condizione di *overflow* di ingresso[25].

Comunicazione seriale

La comunicazione tra software e dispositivo remoto avviene ad una velocità di 38400 bit per secondo. Ogni byte[26] inviato è modificato dal driver seriale in modo da presentare un bit di inizio (start bit), un bit di parità (parity bit), un bit di fine (stop bit) secondo il seguente schema

[23] Con il termine "processo" si intende una istanza di un programma in esecuzione su un determinato elaboratore. Un programma può essere composto da più processi interagenti, tutti generati dallo stesso file eseguibile.

[24] La lettura dei dati acquisiti tramite scheda audio deve essere effettuata a blocchi di dimensione costante. Se il buffer di ingresso non è pieno, la chiamata a funzione `read(...)` rimane in attesa fermando l'esecuzione del processo.

[25] Se il programma che utilizza la scheda audio non è pronto a leggere il *buffer* di ingresso non appena si riempie, i dati ivi contenuti sono scartati e viene generato a schermo (e/o sui log di sistema) un avviso di *overflow*; ma ciò non forza l'interruzione dell'applicazione.

[26] Un byte è composto da 8 bit.

Quindi per ogni byte inviato vengono effettivamente trasmessi 11 bit[27]. Inoltre, secondo il protocollo SONY, ogni comando o risposta scambiato deve avere un byte aggiuntivo contenente il controllo ciclico di errore (CRC[28]).

In corrispondenza del tono di sincronia, l'applicativo deve richiedere il valore attuale del *timecode* ed elaborare la relativa risposta. Con riferimento al protocollo riportato in Appendice A, la richiesta avviene con il comando "61 0C 04", al quale viene aggiunto il byte di CRC. Per un totale di 4 byte, verranno trasmessi 4 * 11 = 44 bit.

Il videoregistratore deve rispondere entro 9 millisecondi con "74 06 xx xx xx xx CRC" (dove "xx" rappresenta un byte di dati), e cioè con 7 byte: in totale sono 77 i bit trasferiti. La sola operazione di comunicazione richiede quindi:

$$\frac{77+44}{38400} + 0.009 \equiv 0.01215\ s = 12.15\ ms \qquad (4.2)$$

4.1.2 Processi concorrenti

L'ordine di grandezza di differenza che esiste tra i due risultati (4.1) e (4.2) potrebbe suggerire di gestire l'intera procedura di ricerca del tono con un singolo processo sequenziale. Ma l'utilizzo di un singolo processo sequenziale si scontra con i seguenti aspetti negativi:

- il sistema operativo Linux è *multitasking* di tipo *time-sharing*. Non è quindi possibile prevedere deterministicamente quando il nostro processo avrebbe accesso alla CPU e per quanto tempo la potrebbe utilizzare, soprattutto in caso di molteplici processi in esecuzione.
- Si possono inoltre verificare errori di comunicazione seriale che richiedono la ripetizione di tutto il comando.

---

[27] L'insieme dei bit di start/stop, eventuale bit di parità e gli 8 bit di dati trasmessi in un secondo viene chiamato "baud". Il baud è l'unità di misura della velocità di trasmissione sulla linea fisica mentre la velocità effettiva dei dati, che si esprime in bit/s., è calcolata come (baud)/(totale bit)*8

[28] Cyclic Redundancy Code. Si calcola sommando i byte di dati e prendendo il byte meno significativo del risultato (se CRC a 8 bit).

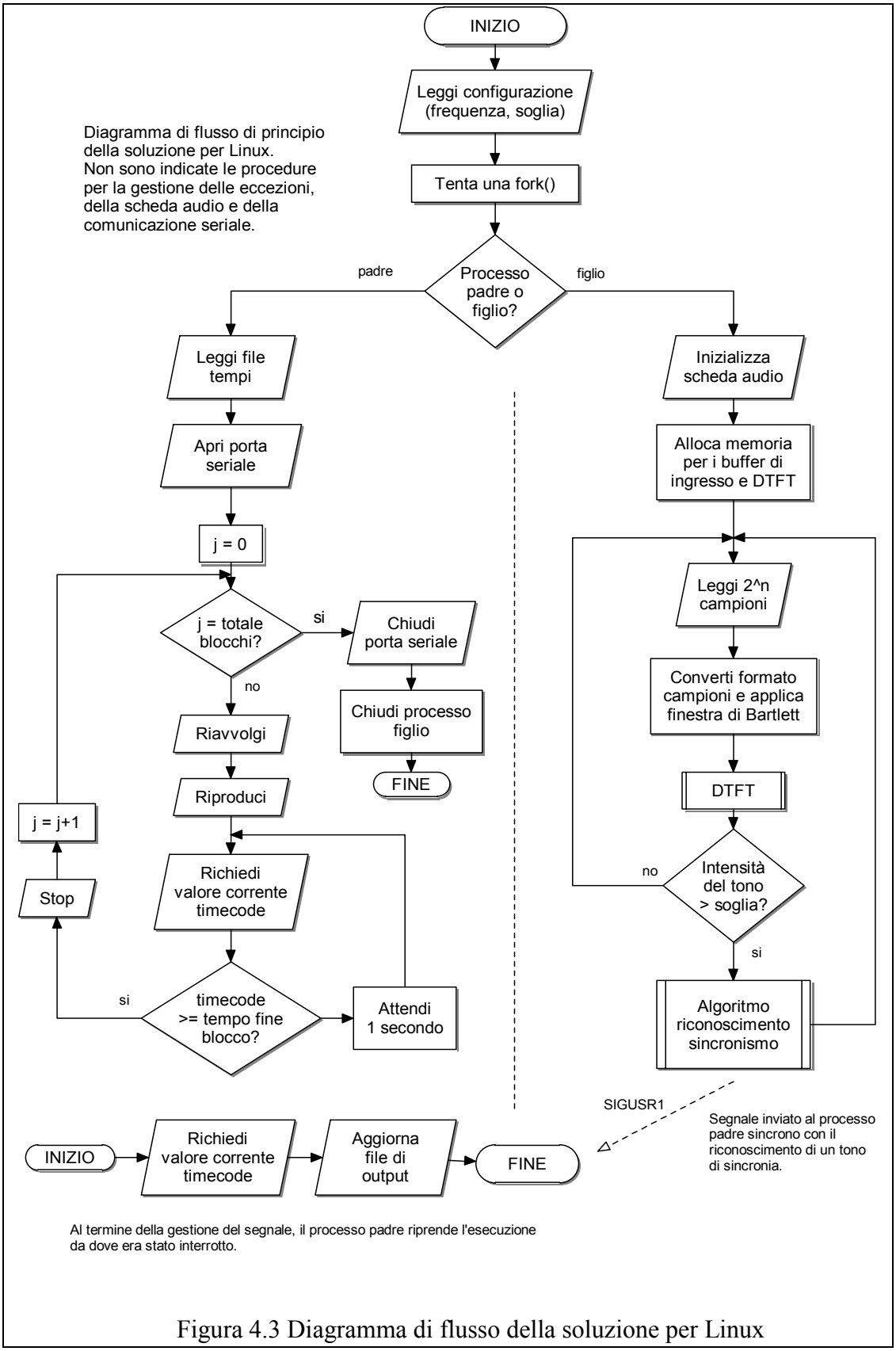

Figura 4.3 Diagramma di flusso della soluzione per Linux

- La distanza temporale tra due comandi inviati al dispositivo remoto deve essere di almeno 10 millisecondi (specifica del protocollo di comunicazione, cfr. Appendice A)

Dunque, in caso di condizioni operative particolarmente sfavorevoli si rischierebbe, con un software sequenziale, di perdere dei dati di sincronia o di richiedere l'intervento dell'operatore.

Il processo "DTFT" viene generato mediante una `fork()` prima di effettuare qualunque inizializzazione (in Figura 4.3 è riportato l'intero diagramma di flusso dell'applicativo in esame). Successivamente i due processi risultanti procedono a verificare la presenza e la compatibilità dell'hardware ai parametri operativi, per poi iniziare la ricerca del segnale di sincronia. La comunicazione tra i due processi avviene solo quando viene individuato il tono di sincronia, mediante l'invio di un segnale utente (`kill(ppid,SIGUSR1,…)`) dal processo DTFT a quello di comunicazione. In caso di errori tali da non poter proseguire l'esecuzione (errori di comunicazione con il videoregistratore, con la scheda audio, tra processi; disco pieno) entrambi i processi vengono terminati.

4.1.3 Linux Open Sound System

Mentre i dispositivi seriali più recenti raggiungono velocità di trasferimento di 38400 baud[29] e superiori, le caratteristiche delle schede audio per PC variano ampiamente da modello a modello. I driver di tipo OSS[30] consentono di modificare tutti i parametri operativi dei dispositivi audio, per essere compatibili con il maggior numero possibile di periferiche sonore (parametri quali frequenza di campionamento, tipo di codifica dei dati campionati, dimensione del buffer di ingresso, numero di canali ed altri ancora).

Al contrario, ogni dispositivo supporta solo un sottoinsieme dei possibili valori dei parametri di funzionamento: è compito del software (e quindi del programmatore) gestire le situazioni in cui la scheda audio, tramite il driver, non può accettare la configurazione impostata. In seguito ad una richiesta di configurazione non supportata

---

[29] Si veda la nota 27

[30] Open Sound System: insieme di driver per schede audio e altri dispositivi sonori sotto sistemi UNIX e compatibili. OSS è stato derivato dal driver audio scritto per il kernel di Linux. La versione attuale funziona su più di una dozzina di sistemi operativi e supporta le schede audio più diffuse così come le periferiche audio integrate nelle schede madre. [57]

il dispositivo imposta la variabile al valore più prossimo a quello richiesto e ne ritorna il valore.

L'applicativo può quindi decidere se procedere con il valore scelto dal dispositivo o se tentare una configurazione diversa. Questo comportamento si presenta spesso per l'impostazione della frequenza di campionamento: la scheda ne realizza fisicamente un numero limitato, i cui valori sono dati dal valore dell'oscillatore installato e dalle possibili configurazioni del divisore di frequenza interno.

Per mantenere la massima compatibilità possibile si è scelta la codifica su 8 bit dei dati campionati (variabili C di tipo `unsigned int`), che è il formato di registrazione supportato dalla maggioranza delle interfacce audio per personal computer [58]. La codifica avviene in PCM, con corrispondenza lineare tra valore reale e valore codificato (al contrario delle quantizzazioni con compressioni di tipo "A" e "μ" che hanno delle corrispondenze non lineari [59], [60]). La frequenza di campionamento può essere impostata dall'operatore, e qualora non fosse supportata dall'hardware, l'applicativo utilizza il valore imposto dalla scheda audio.

I dati letti sono di tipo `unsigned char`, quindi assumono valori tra 0 e 255. In assenza di segnale tutti i campioni hanno valore pari a 127. Si rende necessaria una conversione dei dati in modo da togliere l'offset introdotto dalla scheda audio e poter lavorare su valori di tipo `float`. Nella copia tra il buffer di ingresso e quello della DTFT si applica inoltre la finestra di Bartlett.

```
// copia i dati nel buffer per la FFT
// togliendo l'offset iniziale,
// applicando una finestra di Bartlett e
// convertendo i char in float
for (buf=0;buf<buf_size;buf++) {
        fft_buffer[buf]=((float)audio_buffer[buf]-
127)*(1-fabs(((float)buf-hbuf_size)/hbuf_size));
}
```

Il software fin qui descritto permette la massima flessibilità operativa per la ricerca e il riconoscimento di segnali di sincronia (non necessariamente a frequenza subsonica) in modalità "tempo reale". La compatibilità con le librerie standard del C consente inoltre la compilazione ed il funzionamento su qualunque elaboratore Linux/unix, purché provvisto di porta seriale e scheda audio. Adattando le funzioni che si occupano dell'I/O seriale e audio, il codice della soluzione generale qui presentata compila correttamente su qualunque sistema operativo.

Bisogna però riconoscere che la ricerca del tono implica il raddoppio del tempo di utilizzo del videoregistratore BetaCam, e la conseguente usura delle parti meccaniche che lo compongono. Si è giunti allora ad un applicativo specifico, che si adatta meglio al processo produttivo descritto nel Capitolo 2.

4.2 Applicativo software specifico

Osservando con attenzione i prodotti intermedi del processo di post-produzione (come visto in § 2.1.3), si nota che è altresì possibile intervenire direttamente su uno di essi, senza dover interagire con il sistema di riproduzione ed acquisizione BetaCam.

Vengono così meno la necessità di analisi in tempo reale e il secondo utilizzo del videoriproduttore nella fase di recupero della sincronia. Qui di seguito si motivano le scelte effettuate al fine di giungere all'applicativo software specifico, e si descrivono brevemente i passi implementativi più importanti.

4.2.1 Codifica AVI

L'acquisizione ad alta definizione delle lezioni (dei singoli blocchi) su elaboratore tramite scheda di acquisizione professionale è immagazzinata su file in formato AVI [61]. In un file AVI si possono inserire molteplici flussi sincroni tra loro, di tipo diverso.

Normalmente un file AVI contiene un flusso video ed uno audio; ma si può anche aggiungere tracce MIDI[31], testuali o definite dall'utente. Inoltre lo stesso file AVI può contenere più tracce dello stesso tipo contemporaneamente, anche con formati di codifica diversi (ad esempio MPEG-1 per il video, ADPCM per la prima traccia audio e GSM per la seconda; [63], [64], [65]). Nel caso di flussi di tipo standard (video, audio e MIDI), la riproduzione avviene tramite un lettore multimediale che supporta il formato AVI e il relativo formato di codifica del flusso. Per flussi definiti dall'utente è necessario scrivere il codice apposito.

[31] Musical Instrument Digital Interface: protocollo dedicato alla registrazione e riproduzione di musica su sintetizzatori digitali, supportato da molti produttori di schede audio per personal computer. [62]

I riproduttori multimediali compatibili con il formato AVI ignorano i flussi che non sono in grado di decodificare (ad esempio un lettore audio ignora tutte le tracce di tipo diverso), rendendo così "portabile[32]" il formato AVI.

Date le caratteristiche professionali dell'equipaggiamento utilizzato per digitalizzare le lezioni, è indifferente recuperare il tono di sincronia sull'effettiva registrazione o sulla sua copia informatica. Inoltre il processo produttivo (dall'acquisizione in avanti) avviene con software per Windows: un applicativo costruito appositamente per quest'ultimo sistema operativo semplifica ulteriormente la sincronizzazione tra i flussi audio/video e le relative diapositive. Il file AVI come prodotto intermedio di processo contiene una singola traccia sonora, la quale riporta tutto l'audio prodotto durante la registrazione (uscita del miscelatore audio), codificata in PCM stereo a 44.1 kHz e con 16 bit per campione (cfr. § 2.1.3). Quest'ultimo è il formato cui si è fatto riferimento nello sviluppo della soluzione specifica.

L'applicativo realizzato e testato, che rappresenta il risultato finale di questa tesi, scorre la traccia audio dei file AVI risultanti dall'acquisizione dei blocchi delle lezioni, ne effettua l'analisi spettrale alla ricerca del tono di sincronia e genera un file contenente i tempi di cambio diapositiva riferiti all'inizio del blocco. Non è più necessario interagire con il riproduttore BetaCam ed analizzare in tempo reale il contenuto spettrale della traccia audio come visto per la soluzione generica (cfr. § 4.1). In Figura 4.4 è rappresentato un diagramma di flusso di principio per la ricerca di un tono in un file AVI.

Per facilitare la definizione del livello di soglia l'applicativo propone, prima di ogni ricerca, i valori minimi e massimi della componente spettrale del tono riscontrati nel file AVI sotto analisi.

---

[32] La terminologia inglese indica con "portable" la possibilità di poter riutilizzare delle componenti software su molteplici sistemi operativi. Un codice sorgente "portabile" genera un programma eseguibile su più piattaforme. Quindi un file AVI è riproducibile su qualunque sistema operativo, a condizione che esista il software adatto allo scopo.

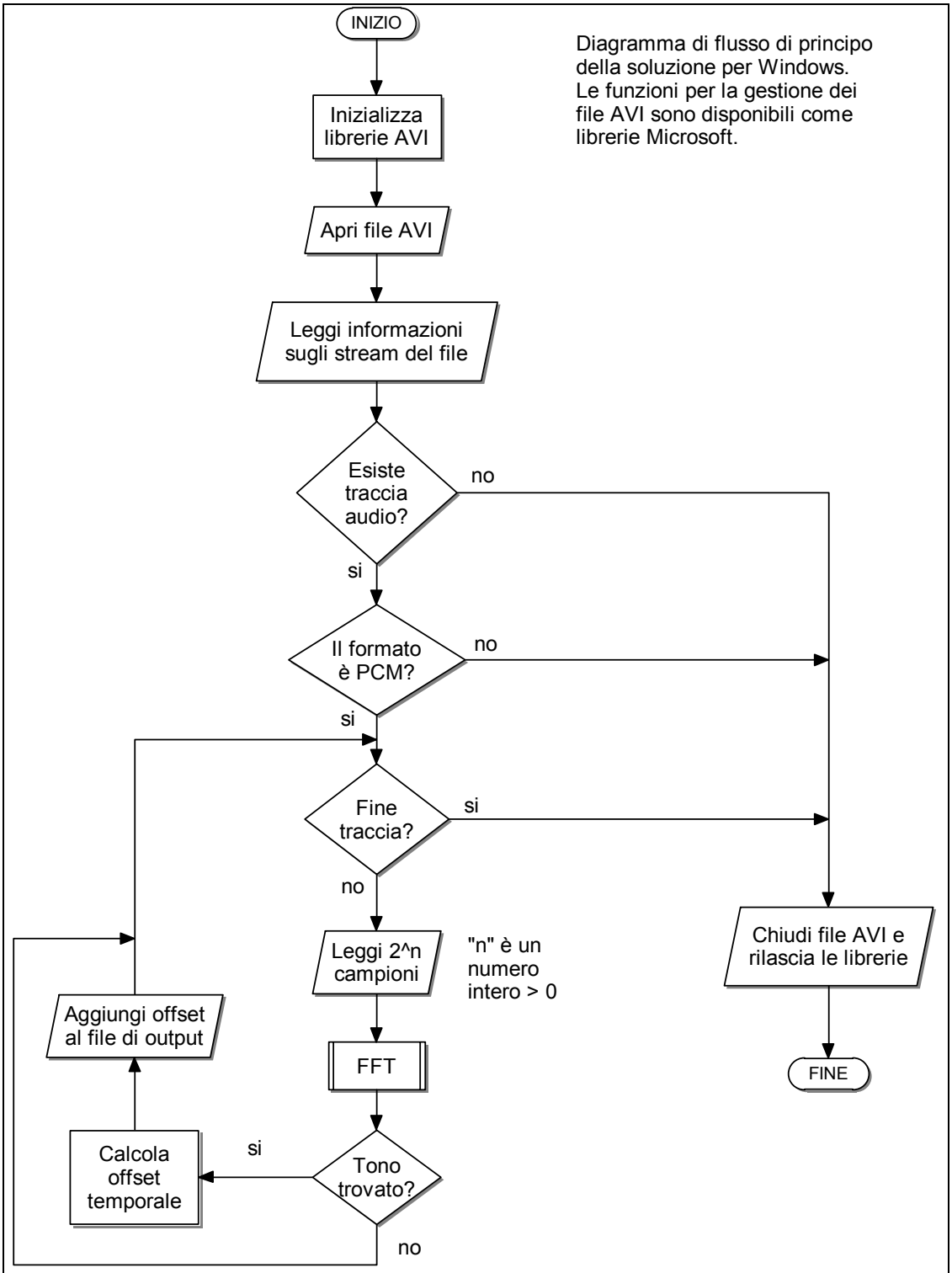

Figura 4.4 Diagramma di flusso di principio della soluzione specifica

4.2.2 Librerie MS AVIFile

Le API Microsoft documentate da MSDN (Microsoft Developer Network [66]) permettono di leggere e scrivere un file AVI operando ad alto livello, senza dover

gestire i flussi multimediali come effettivamente descritti e immagazzinati nel formato RIFF (file con contenuto rappresentabile su una base tempi). Al fine di massimizzare la velocità di esecuzione dell'intera ricerca del tono di sincronia il programma è stato scritto in C sfruttando quanto più possibile le funzioni di libreria (Microsoft *Visual Studio 6*). La descrizione delle funzioni utilizzate per l'implementazione della soluzione proposta è qui riportata per documentare l'esatto ordine con cui esse si richiamano.

Per utilizzare le librerie AVIFile di Microsoft è necessario provvedere alla loro inizializzazione prima dell'utilizzo chiamando

```
AVIFileInit();
```

senza parametri e senza valore di ritorno. Le funzioni contenute in questa libreria dinamica (DLL) gestiscono le informazioni dei file RIFF come uno o più flussi di dati separati. Un applicativo software che usa le librerie AVIFile può accedere separatamente ai vari flussi.

L'apertura di un file AVI si effettua con

```
AVIFileOpen(&pfile, szFile, OF_SHARE_DENY_WRITE, 0L);
```

I parametri di chiamata sono, in ordine: il puntatore al buffer di interfaccia con il file (`pfile`), il nome del file, il modo di accesso e il tipo di file da gestire.

Se l'operazione di apertura del file non ritorna errori, si devono identificare e immagazzinare in un vettore (`gapavi[]`) i puntatori ai flussi ivi contenuti.

```
AVIFileGetStream(pfile, &gapavi[i], 0L, i)
```

Da questo punto il software ha accesso ai flussi tramite i puntatori registrati in `gapavi[]` e non più con il puntatore al file `pfile`. Quindi, per questa specifica applicazione, si cerca la presenza della traccia AVI di tipo audio (`streamtypeAUDIO`) scorrendo tutti i flussi del file:

```
for (i = gcpavi; i < MAXNUMSTREAMS; i++) {
        gapavi[i] = NULL;
        if (AVIFileGetStream(pfile, &gapavi[i], 0L, i -
gcpavi) != AVIERR_OK) break;
        if (gapavi[i] == NULL) break;
}
gcpavi = i; // gcpavi è il totale dei flussi trovati
```

`gcpavi` rappresenta il numero totale di flussi contenuti nel file, e il valore della variabile `i` qui sotto rappresenta l'identificativo del puntatore al flusso audio in `gapavi[]`:

```
for (i = 0; i < gcpavi; i++) {
        AVIStreamInfo(gapavi[i], &avis, sizeof(avis));
        if (avis.fccType == streamtypeAUDIO) {
                // ...
                // funzioni per l'elaborazione dell'audio
                // ...
                return i;
        }
}
```

`avis` è una struttura di tipo AVISTREAMINFO (riportata in Appendice B1), e viene associata alle informazioni relative al flusso dati tramite `AVIStreamInfo()`. In base alle informazioni ricavate da questa struttura si verifica se il flusso è di tipo audio e si deduce la dimensione dei campioni, necessaria ad interpretare correttamente gli stessi dati (Figura 4.5).

```
// Per la codifica PCM su 16 bit ogni campione
// è composto da:
// - byte 1 e 2: parte bassa e alta del canale sinistro
// - byte 3 e 4: parte bassa e alta del canale destro
// Se il flusso e' mono, "sample size" e' la meta'
// del caso stereo (2 invece di 4), e i campioni
// sono affiancati.
slSampleSize = (LONG)avis.dwSampleSize;
```

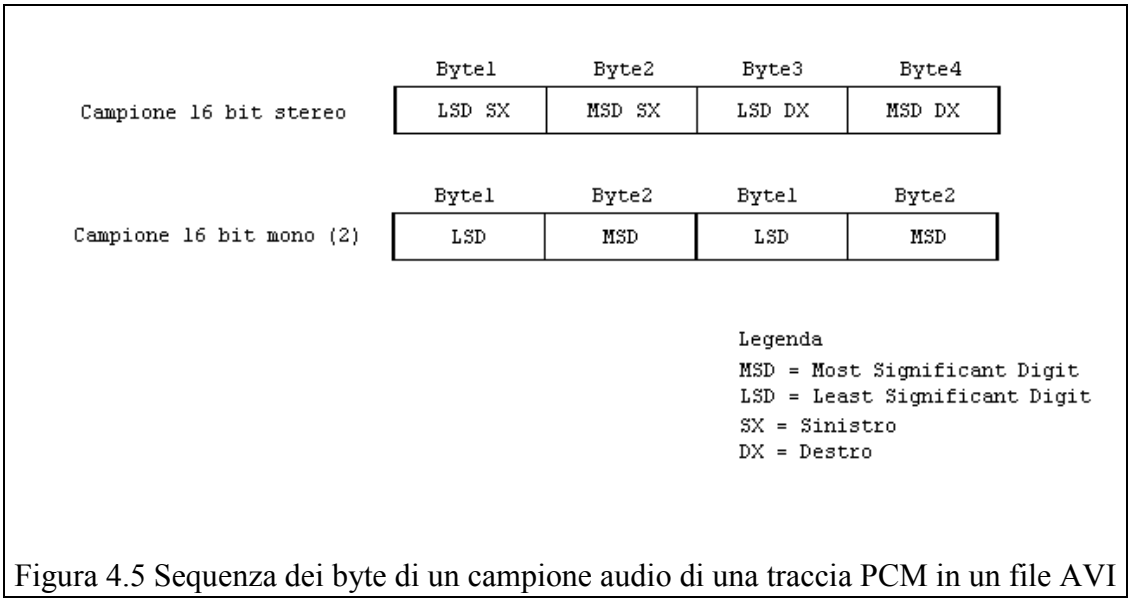

Figura 4.5 Sequenza dei byte di un campione audio di una traccia PCM in un file AVI

Scelto il flusso audio su cui operare (qualora ce ne fosse più di uno, il software realizzato utilizza il primo trovato) si inizializza la struttura `bi` di tipo WAVEFORMATEX con il formato dei dati della traccia audio:

```
AVIStreamReadFormat(gapavi[i], 0, &bi, &lStreamSize);
```

La struttura di tipo WAVEFORMATEX è descritta nella Appendice B2.

`NBR_OF_SAMPLES` è definito come potenza intera di 2 (2^n). Quindi si alloca dinamicamente la quantità di memoria necessaria per leggere `NBR_OF_SAMPLES` campioni, ciascuno costituito da bi.nBlockAlign byte (un `char` occupa un byte). Analogamente si alloca la memoria da assegnare al buffer su cui lavora la funzione di DTFT, il quale deve poter contenere `NBR_OF_SAMPLES` valori di tipo `float`.

```
// Alloca memoria per leggere un blocco
if ((lpOld = (char *)GlobalAlloc(0, bi.nBlockAlign *
    NBR_OF_SAMPLES * sizeof(char))) == NULL) { ... }
// Alloca memoria per il buffer FFT
if ((fft_buffer = (float *)GlobalAlloc(0, NBR_OF_SAMPLES
    * sizeof(float))) == NULL) {
```

Infine si può scorrere tutta la traccia audio AVI, spostando in `lpOld` dati in blocchi di `NBR_OF_SAMPLES` campioni alla volta:

```
AVIStreamRead(gapavi[i], lStreamSize, NBR_OF_SAMPLES,
    (LPVOID)lpOld, bi.nBlockAlign * NBR_OF_SAMPLES,
    &lBytes, &lSamples);
```

La chiamata ad `AVIStreamRead()` informa inoltre l'applicativo sul numero di byte (`lBytes`) e campioni (`lSamples`) effettivamente trasferiti. Qualora il numero di campioni scritti nel buffer `lpOld` sia inferiore a `NBR_OF_SAMPLES` si rende necessario uno *zero padding*[33] per avere dati validi su cui effettuare l'analisi spettrale.

Considerato il formato con cui i campioni PCM sono rappresentati nella traccia AVI si rende necessaria una operazione matematica di conversione. Alla conversione del campione in `float` (formato richiesto dalla procedura di DTFT) si affianca l'applicazione della finestra di Bartlett.

```
for (j=0;j<(lSamples*bi.nBlockAlign);j=j+slSampleSize) {
        // Converti i campioni in (float)
        // e applica una finestra di Bartlett
        // Il formato dei dati a 16 bit è
```

[33] Si inseriscono, in coda ai dati letti, tanti valori nulli quanti sono necessari a riempire il buffer.

```
        // * lpOld[j]   : LSD (-128/127)
        // * lpOld[j+1] : MSD (-128/127)
        // Quindi serve una conversione:
        // float = MSD*256 + (LSD+128)
        // Dato che si usa solo il canale sinistro
        // non considera lpOld[j+2] e [j+3]
        tmpsample= (float)(lpOld[j+1]*256 + lpOld[j] +
128);
        fft_buffer[j/bi.nBlockAlign] = tmpsample * (1-
fabs(((j/bi.nBlockAlign)-
HNBR_OF_SAMPLES)/HNBR_OF_SAMPLES));
} // end for
```

A questo punto si può calcolare lo spettro dei campioni letti:

```
realft(fft_buffer-1,NBR_OF_SAMPLES,1);
```

La funzione `realft(...)` opera una trasformata di Fourier di `NBR_OF_SAMPLES` (2^n) dati reali ed è descritta in 3.4 Riconoscimento segnale di sincronia. Il contenuto di `fft_buffer` viene sostituito con l'immagine spettrale (solo per le frequenze positive) dei `NBR_OF_SAMPLES` campioni analizzati, con i valori reali e immaginari alternati (…, Re[n], Im[n], Re[n+1], Im[n+1], …). In `fft_buffer` ci sono `NBR_OF_SAMPLES`/2 campioni di frequenza. Di conseguenza, detta `freq` la variabile contenente il valore della frequenza del tono di sincronia, il valore che ne rappresenta la densità spettrale si trova nella posizione my_n del vettore di uscita della FFT:

```
my_n = 2 * (int)floor(NBR_OF_SAMPLES * freq /
    bi.nSamplesPerSec);
```

Una valutazione qualitativa dell'ampiezza della riga più prossima a `freq` nello spettro calcolato si ottiene dalla radice quadrata della somma del quadrato delle componenti reali e immaginarie:

```
my_f_m = sqrt(fft_buffer[my_n]*fft_buffer[my_n] +
    fft_buffer[my_n+1]*fft_buffer[my_n+1]);
```

Confrontando `my_f_m` con un valore di soglia prestabilito si decide sulla presenza del tono. Le prove effettuate hanno mostrato incrementi di almeno quattro ordini di grandezza per `my_f_m` in presenza del tono: si conferma così che le misure assolute di ampiezza non sono necessarie.

In ultimo si deve avere l'accortezza di chiudere il file analizzato e rilasciare le librerie AVIFile:

```
AVIFileRelease(pfile);  // chiude il file
```

```
AVIFileExit();          // rilascia la libreria AVIFile
```

La compilazione del progetto relativo al software descritto sopra è stata eseguita con i *tool* di sviluppo Microsoft Visual Studio 6.0. Le funzioni di alto livello per la gestione degli AVI sono dichiarate in `vfw.h`. La compilazione e il successivo *linking* devono comprendere la libreria statica `vfw32.lib`.

L'applicativo specifico qui descritto non opera in tempo reale, di conseguenza non richiede il rispetto di stringenti requisiti temporali. La velocità di esecuzione è direttamente proporzionale alla potenza di calcolo dell'elaboratore utilizzato come illustrato nel capitolo seguente.

5 Analisi dei risultati

5.1 Influenza sul prodotto finito

Una eventuale influenza del tono a frequenza subsonica sul prodotto finito si può manifestare in tre modi: saturazione dell'audio utile, impulsi in concomitanza dell'inizio e fine del tono di sincronia, udibilità del tono di sincronia. I disturbi possono anche verificarsi contemporaneamente e sono stati riprodotti sperimentalmente.

La distorsione di saturazione avviene quando il tono sinusoidale ha una ampiezza prossima a quella massima consentita dal sistema di registrazione. La somma, nel tempo, dell'ampiezza del tono con quella dell'audio utile porta ad una "tosatura" dell'audio, e produce una fastidiosa distorsione percepibile da un ascoltatore.

Gli impulsi sono dovuti al cambio repentino di ampiezza del segnale audio provocato dall'introduzione e rimozione di un tono ad ampiezza costante. Sperimentalmente si è verificato che introducendo una evanescenza sull'ampiezza del tono, all'inizio e alla fine dello stesso (Figura 5.1), gli impulsi scompaiono. Sempre sperimentalmente, si è valutato in un decimo di secondo il tempo di salita dell'ampiezza da 0 al valore massimo. Il risultato è il medesimo per la durata dell'evanescenza di uscita.

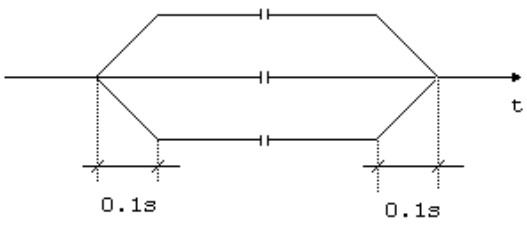

Figura 5.1 Andamento suggerito dell'inviluppo di ampiezza del tono subsonico per eliminare gli impulsi altrimenti udibili nel prodotto finito

Le prove effettuate inserendo via software dei toni di sincronia in una traccia AVI complessa (voce, musica e rumore di fondo contemporanei) hanno dato dei risultati molto positivi: lo scarto tra il tempo di inizio del segnale di sincronia e il tempo calcolato dall'applicativo specifico è dell'ordine del decimo di secondo (in più o in meno). Considerando la pausa di alcuni secondi che il docente effettua in

corrispondenza del cambio della *slide* lo scarto non penalizza l'utente del prodotto finito. Lo scarto è di durata paragonabile alla durata dei campioni su cui si effettua la DTFT: un intervallo di analisi più lungo comporta minore accuratezza dei risultati (cfr. paragrafo successivo)..

Per mancanza di strumenti di misura non è stato possibile verificare la cancellazione del segnale di sincronia da parte dell'algoritmo di compressione MPEG sulla base delle considerazioni *psicoacustiche* viste nel Capitolo 3. Il tono non è comunque rilevabile nella riproduzione del prodotto finito.

5.2 Parametri di ricerca

La scelta della frequenza del tono subsonico è stata fatta ascoltando la riproduzione di varie frequenze ad alto volume, considerando le bande passanti dei dispositivi di registrazione e le caratteristiche uditive medie dell'orecchio umano. L'intervallo di valori accettabili è compreso tra il valore più alto dei limiti minimi delle bande passanti e il valore di frequenza per cui il tono diventa udibile:

$$20\ \text{Hz} \leq \text{frequenza tono} \leq 70\ \text{Hz}$$

Inoltre, nell'intervallo di frequenze considerato, una riproduzione ad alto volume viene percepita più a livello fisico che a livello uditivo [67].

Per gli effetti dell'espansione delle componenti spettrali ad opera del *windowing* non si è tenuto conto della relazione che esiste tra la frequenza di campionamento e la discretizzazione dello spettro di frequenza calcolato con una DTFT. Di conseguenza la frequenza scelta può ricadere liberamente nell'intervallo definito sopra. La frequenza adottata nelle prove sperimentali è di 35 Hz.

Al fine di massimizzare l'efficienza computazionale si è scelto di utilizzare un algoritmo di tipo DTFT che ha complessità dell'ordine di $N*\log_2(N)$, contro N^2 della procedura generica di analisi di Fourier (N = lunghezza della DTFT, o numero di campioni analizzati per volta). Ciò comporta l'elaborazione di una sequenza di N campioni sempre lunga 2^n (con n intero e maggiore di zero). Se il numero di campioni disponibili non è una potenza intera di 2, si rende necessario effettuare uno *zero padding* fino a raggiungere il valore di 2^n più prossimo.

Particolare attenzione va prestata comunque alla distanza tra le frequenze discrete per non aumentare, a causa del *windowing*, la probabilità di interferenze nell'algoritmo di riconoscimento della sincronia provocate da componenti spettrali con frequenza relativamente prossima al tono inserito artificialmente. La distanza tra le frequenze discrete si calcola come:

$$\delta F = \frac{1}{N \times \Delta} = \frac{f_c}{N}$$

con N pari alla lunghezza della DTFT, Δ pari all'intervallo di campionamento e f_c frequenza di campionamento. Scegliendo N di 4096 campioni e Δ di 44100^{-1} Hz ≅ 22.67 μs:

$$\delta F = \frac{44100\ bit/s}{4096\ bit} \cong 10.76\ Hz$$

Così per un tono di 35 Hz, valore che rientra nella stima della frequenza discreta di 32.28 Hz (la più prossima), utilizzando una finestra di Bartlett per modificare i campioni si potrebbero riscontrare, attenuate in varia misura, le riproduzioni delle componenti spettrali fino a 2*δF frequenze adiacenti, cioè comprese tra

$$32.28 - 2 \times 10.76 = 10.76\ Hz$$

e

$$32.28 + 2 \times 10.76 = 53.8\ Hz$$

Eventuali musiche inserite nella traccia audio (che hanno componenti spettrali che scendono anche fino a 50 Hz) potrebbero interferire con il tono di sincronia qualora la sua frequenza base rientrasse in intervalli di frequenza più alti: pertanto non si consiglia l'adozione di un tono di sincronia con frequenza prossima a 70 Hz se la registrazione contiene contributi musicali.

La probabilità di segnali utili interferenti nel campo delle frequenze subsoniche influenza la scelta del valore di soglia utilizzato dall'algoritmo di riconoscimento, così come la durata del tono di sincronia.

fc [Hz]	N	dF [Hz]	t [s]
8192	4096	2	0,5
8192	8192	1	1
11025	4096	2,69165	0,371519
11025	8192	1,345825	0,743039
22050	4096	5,383301	0,18576
22050	8192	2,69165	0,371519
44100	4096	10,7666	0,09288
44100	8192	5,383301	0,18576

Tabella 5.1 Distanza tra le frequenze discrete δF e velocità di produzione di N campioni, in funzione della frequenza di campionamento f_c e della lunghezza N della DTFT

La scelta della distanza tra le frequenze discrete presenta due gradi di libertà: la frequenza di campionamento f_c e la lunghezza N della DTFT (Tabella 5.1). Mentre per la prima deve essere scelto il valore più alto possibile per minimizzare gli effetti dell'*aliasing* (44100 Hz), la determinazione della lunghezza N della DTFT deve tenere conto di:

- velocità computazionale dell'elaboratore per analisi in tempo reale
- risoluzione temporale nella determinazione dell'instante di sincronia

A parità di frequenza di campionamento, un innalzamento di N diminuisce la richiesta di velocità all'elaboratore e contemporaneamente migliora la risoluzione in frequenza. Sebbene entrambi gli effetti riscontrati siano positivi, una migliore risoluzione in frequenza comporta una peggiore risoluzione temporale, e quindi una minore attendibilità dei risultati nella ricostruzione della sincronia.

Si è scelto, dove possibile, di utilizzare una frequenza di campionamento di 44100 Hz e 4096 campioni per singola analisi spettrale. Combinazioni dei parametri differenti da quella suggerita non precludono il corretto funzionamento della procedura di ricerca e riconoscimento del tono di sincronia.

Come ultimo parametro da determinare per una ricerca efficace ed affidabile del tono di sincronia, vi è la durata temporale del tono stesso. Tale valore è funzione della risoluzione temporale della DTFT vista sopra, cioè dell'effettiva durata di un gruppo di campioni, e allo stesso tempo della rumorosità del sistema di registrazione e riproduzione. In seguito alla scelta dei parametri della DTFT le caratteristiche del tono sono le seguenti:

Durata minima	$10 \times \frac{4096}{44100} \cong 928\ ms$
Durata massima	senza limite
Distanza tra toni	$\frac{4096}{44100} \geq 92.8\ ms$

La durata minima scelta è di 10 intervalli di analisi DTFT: la componente spettrale associata alla frequenza del tono di sincronia deve essere superiore al valore di soglia per dieci analisi DTFT consecutive.

Va inoltre considerato che le evanescenze di inizio e fine tono introducono un transitorio di almeno 2*0.1 = 0.2 secondi, che portano la durata minima totale a

$$0.928 + 0.2 = 1.128 \text{ s}$$

5.3 Fattore economico

Come la maggior parte delle innovazioni tecnologiche, anche quella qui descritta ha un'influenza positiva sui costi sostenuti dai suoi utilizzatori (produttori di materiale multimediale)

Misurare con esattezza il beneficio apportato dall'introduzione di un software per recuperare il sincronismo nella produzione di lezioni multimediali non è un compito facile, e probabilmente può avvenire solo dopo un utilizzo prolungato dell'applicativo. Tra le variabili vi sono il fattore umano (facilità con cui l'operatore si adatta al nuovo procedimento e lo mette in opera) e i vari fattori tecnologici (principalmente la potenza di calcolo dell'elaboratore utilizzato per il recupero del sincronismo).

Da notare che, per sopperire alla variabile del fattore umano, ove l'operatore presentasse difficoltà ad impadronirsi della nuova tecnica, potrebbe essere opportuna una rapida formazione sul campo (durata orientativa 30 minuti) e/o la redazione di un foglio di istruzioni operative (*quick reference*).

Peraltro quantificare i vantaggi economici solo in termini di ore-uomo risparmiate è riduttivo: un processo produttivo più rapido consente di aumentare il numero di lezioni prodotte, e di conseguenza presentarsi sul mercato con una offerta maggiore, il che la quale dovrebbe significare entrate più elevate.

Si tenta ora una stima delle ore uomo risparmiate sulla base delle tempistiche rilevate durante i test.

Ipotizzando in 45 minuti la durata media di una lezione, i singoli passi della produzione senza l'automatismo qui sviluppato richiedono (in corsivo sono le operazioni che non necessitano della presenza di un operatore):

1. 1 ora per la registrazione
2. *45 minuti per l'acquisizione su computer (da BetaCam ad AVI)*
3. 45 minuti per il recupero della sincronia con le diapositive
4. 30 minuti per il montaggio video
5. *12 ore (medie) per la compressione e conversione in ASF (da AVI ad ASF)*
6. 15 minuti per l'adattamento dell'interfaccia utente

per un totale di circa 15 ore. Escludendo i passi eseguiti in automatico senza la presenza di un operatore, le ore-uomo necessarie sono 2 ore e 30 minuti.

Sostituendo il terzo passo di cui sopra con la soluzione specifica proposta, il recupero della sincronia richiede un tempo variabile tra 1 e 10 minuti (in funzione del numero dei blocchi che costituiscono la lezione). Assumendo una media di 5 minuti a lezione, si ha un risparmio di 45' - 5' = 40'. Ne consegue che il tempo totale di produzione che richiede la presenza di un operatore si riduce a 1 ora e 50 minuti (27% di tempo risparmiato).

Proseguendo ulteriormente nella stima, va ancora considerato che l'addetto alla registrazione (regista) non si occupa dei passi successivi: la registrazione di una lezione e la post-produzione di un'altra possono coesistere. In tal modo il risparmio percentuale si alza al 45%:

$$\frac{\textit{tempo risparmiato}}{\textit{tempo originale di produzione}} = \frac{40'}{45'+30'+15'} = \frac{40'}{90'} \cong 0.45 \equiv 45\%$$

Si può concludere che, considerando solo i passi assistiti da operatore nelle operazioni di post-produzione, si potranno produrre quasi 2 lezioni laddove se ne produceva una sola prima dell'introduzione dell'automatismo qui sviluppato.

5.4 Prestazioni

Si procede ora ad analizzare le *performance* dei due applicativi software, in funzione dell'hardware su cui sono stati testati.

5.4.1 Applicativo generico

Trattasi del software in ambiente Linux che dialoga con il riproduttore BetaCam ed analizza in tempo reale la traccia audio. E' stato testato su un computer Intel 486 DX a 66 MHz, 8 Mbyte di RAM. La scheda audio, basata su processore CMI8330, è risultata molto rumorosa soprattutto alle basse frequenze, dove sembra convertire in segnali audio i campi elettromagnetici prodotti all'interno del computer stesso. Stesso comportamento si è verificato utilizzando una scheda basata su processore ESS1868 e una *Creative Sound Vibra 16*. Ciononostante il riconoscimento del segnale di sincronia a 35 Hz avviene senza disturbi e senza apprezzabili scarti temporali.

La potenza di calcolo offerta dal processore 486 DX/66 è sufficiente ad analizzare in tempo reale dati campionati a 44.1 kHz suddivisi in gruppi da 4096 byte (lunghezza della DTFT), e contemporaneamente dialogare con il videoregistratore BetaCam.

La compilazione del codice sorgente richiede circa 10 secondi con il compilatore `gcc` e il file eseguibile risultante è di 30 kbyte circa. Qualora fosse necessaria l'inclusione in modo statico delle librerie utilizzate nel file eseguibile, la dimensione sale a 1.5 Mbyte circa.

I parametri operativi vanno passati all'applicativo tramite riga di comando ad esclusione del file contenente i tempi di inizio e fine dei singoli blocchi costituenti la lezione.

5.4.2 Applicativo specifico

L'applicativo specifico funziona su piattaforme Windows e il suo *input* è un file AVI. Il test di velocità è avvenuto su una macchina con Windows NT, equipaggiata con un processore Intel Pentium III a 450 MHz. In condizioni di macchina "scarica" (senza altri processi eseguiti contemporaneamente che richiedano un uso intensivo della CPU) l'analisi di un blocco di 5 minuti di lezione richiede circa 10 secondi.

La compilazione del codice sorgente è stata effettuata con Microsoft Visual C++ 6.0 in 20 secondi circa; questa durata è indifferente dalla potenza di calcolo del computer utilizzato (test effettuato su processori Pentium 166 MMX e Pentium II 200). L'operazione di *link* deve includere anche la libreria statica `vfw32.lib` (da specificare nei "Settings" del "Project" alla voce "Link") che contiene le funzioni ad alto livello per la gestione del formato AVI.

Il risultante file eseguibile presenta una interfaccia di tipo testuale (DOS) ed ha dimensione di 200 kByte circa. I parametri operativi devono essere inseriti da tastiera in modo interattivo dall'operatore (nome del file AVI, frequenza del tono cercato, livello di soglia).

6 Conclusioni

I risultati ottenuti con le verifiche sul campo hanno confermato la fondatezza delle ipotesi fatte per poter applicare la tecnica *steganografica* proposta (cfr. § 3.2.4).

Il recupero della sincronia avviene automaticamente con gli applicativi sviluppati e con una precisione difficilmente raggiungibile da un operatore. Lo sgravio di quest'ultimo da un lavoro ripetitivo consente inoltre di velocizzare il processo produttivo di materiale multimediale distribuibile tramite Internet o CD-ROM. Con riferimento alle prove effettuate, previa calibrazione del livello di soglia, il riconoscimento del tono subsonico avviene con affidabilità prossima al 100%.

I parametri operativi adottati per la ricerca e il riconoscimento del tono di sincronia (quali lunghezza temporale di un segnale valido, lunghezza della DTFT e frequenza di campionamento) non sono critici. Rispettando le dipendenze viste in § 5.2, il recupero della sincronia è affidabile anche con valori diversi da quelli descritti in questa sede.

La frequenza del tono è compresa in un intervallo relativamente esteso (20 ÷ 70 Hz), che permette ampio margine operativo in funzione delle bande passanti dei sistemi coinvolti nella produzione. Per l'intervallo di frequenze accertato nel corso delle prove operative, valgono inoltre le considerazioni *psicoacustiche* descritte nel paragrafo 3.2.3: il tono non è percepibile nel prodotto finito.

Entrambi gli applicativi non richiedono particolari configurazioni hardware e/o software, conferendo estrema flessibilità al prodotto realizzato. Un eventuale sottodimensionamento della potenza di calcolo dell'hardware grava sulle prestazioni ma non sulle funzionalità degli applicativi.

L'automatizzazione di un passo della post-produzione di lezioni e/o presentazioni multimediali è realtà.

La soluzione *steganografica* proposta non preclude ulteriori sviluppi degli applicativi descritti in campi applicativi similari.

6.1 Sviluppi futuri

La procedura attuale per la produzione delle lezioni multimediali (cfr. Capitolo 2) comporta l'utilizzo di *slide* statiche, senza che i vari punti discussi nella *slide* possano apparire uno per volta. Il campo d'applicazione della tecnica *steganografica* qui proposta si può estendere a tutte le applicazioni in cui un flusso audiovisivo deve essere sincronizzato con delle immagini computerizzate, statiche o no. Condizione

fondamentale per l'utilizzo delle soluzioni software proposte è che si possa generare il tono subsonico in sincronia con gli eventi della presentazione (nuova *slide*, nuova riga di testo, …).

Una eventuale combinazione di frequenze subsoniche diverse (contemporanee o in sequenza) permette di associare alla traccia audio delle notificazioni di sincronia a maggiore contenuto informativo, conferendo ulteriore flessibilità agli applicativi proposti in questa sede. Tali informazioni potrebbero, ad esempio, diversificare il tipo di azione da intraprendere tra un ventaglio di possibilità (caricamento pagina Web, aggiornamento *slide*, effetto sonoro, …).

Considerando invece l'attuale assenza dal mercato di software per la creazione di prodotti multimediali che permettano la sincronizzazione a posteriori di flussi multimediali, è opportuno ipotizzare di sviluppare ulteriormente i risultati qui ottenuti. Mantenendo la caratteristica di software multi-piattaforma, sarebbe auspicabile arrivare all'allineamento delle sue funzionalità sia per Windows sia per Linux.

L'allineamento dovrebbe consentire una sincronizzazione dei flussi audio/video con quello testuale, sia partendo da un file in formato AVI (applicativo specifico, cfr. § 4.2) sia interagendo direttamente con il sistema di acquisizione (lettore Betacam con supporto del protocollo di comunicazione seriale SONY, applicativo generico, cfr. § 4.1).

In ambiente Linux sono in fase di sviluppo nella comunità *Open Source* le librerie Avifile [68]. Esse consentono la gestione, in linguaggio C, del formato AVI sia in lettura sia in scrittura. Prima di procedere ad un loro utilizzo per l'applicativo proposto sarà opportuno valutare la loro affidabilità e conformità alle specifiche operative richieste.

In ambiente Windows si rende invece necessaria la personalizzazione del codice per la comunicazione seriale (secondo il protocollo SONY) e la gestione delle periferiche sonore.

Appendice A Sony 9-Pin Remote Protocol

Documentazione completa del protocollo Sony per la comunicazione con videoregistratori BetaCam. Autore: Rick Davies (rld@well.com).

Sony 9-Pin Remote Protocol

This is not the official Sony 9-pin protocol. It is a summary for reference purposes only. To obtain the protocol document, contact Sony directly.

Updated 6/16/96. Fixed the Prog Speed formula. Added Tables. Added all group 20 descriptions, more info on status bits, and most of the group 40 commands.

Communication Format

The protocol is based on the EIA RS-422-A signal standard, usually at 38.4 kBit/s. The data are sent as 1 start bit + 8 data bits + 1parity bit + 1 stop bit. Parity is odd: the bitwise sum of data bits 0 -7 and the parity bit is an odd number.

Command Block Format

The controlling device and the controlled device communicate through the interchange of command blocks. The bytes in each command block are assigned as follows:

- **CMD-1/DATA COUNT**. **CMD-1** is the upper 4 bits, **DATA COUNT** is the lower 4.
- **CMD-2**.
- **DATA-1** up to **DATA-N**, where n is the value in data count
- **CHECKSUM**

CMD-1

Indicates the function and direction of the command, according to:

0	System control (Master->Slave)
1	Return for 0,2, or 4 of cmd-1 (Slave->Master)
2	Transport Control (Master->Slave)
4	Preset/Select control (Master->Slave)
6	Sense Request (Master->Slave)
7	Sense Return (Slave->Master)

DATA COUNT

Indicates the number of bytes (max 15) inserted between **CMD-2** and **CHECKSUM**

CMD-2

Designates the command. Refer to the command table for definitions. Ex. **CMD-1**=0 and **CMD-2**=0C means LOCAL DISABLE.

DATA-1 to **DATA-N**

Data which correspond to those indicated by the command. Refer to the command table for data formats.

CHECKSUM

Lower eight bits of the sum of the bytes in the command block.

Communication Protocol

The protocol is initiated by the master. The slave should return a response within 9 msec. The response may be:

- NAK + Error Data: Undefined command or communication error
- COMMAND + Data: if Command requested data
- ACK: if Command did not request data

The master should not send another command until receiving a response from the slave device. The master must also insure that no more than 10 msec lapses between bytes in a command block. The master must immediately stop sending data when it receives a NAK + Error Data message. If the Error Data contains "Undefined Command" the master may immediately send another command, otherwise it must wait at least10 msec before sending another command. When the master does not receive a response from the slave within the 10 msec timeout, it may assume that communications have ceased and take appropriate measures.

Cabling

The pin assignments for the 9-pin cable are as follows:

Pin	Master	Slave
1	Ground	Ground
2	Rcv A	Xmit A
3	Xmit B	Rcv B
4	Xmit Common	Rcv Common
5	Spare	Spare
6	Rcv Common	Xmit Common
7	Rcv B	Xmit B
8	Xmit A	Rcv A
9	Ground	Ground

Auth - this varies a lot: contact individual manufacturers for pinouts.

Command Table

This is the command table for the DVR-2000/2100. It is summarised here for reference purposes only. If you want to order it from Sony call the Kansas City supply center and hope you get someone who knows the part number for the recorder whose protocol you want to use.

	Command		Response
00 0C	Local Disable	10 01	Ack
00 11	Device Type Request	12 11	Device Type
00 1D	Local Enable	10 01	Ack
20 00	Stop	10 01	Ack
20 01	Play	10 01	Ack
20 02	Record	10 01	Ack
20 04	Standby Off	10 01	Ack
20 05	Standby On	10 01	Ack

20 0F	Eject	10 01	Ack
20 10	Fast Fwd	10 01	Ack
2X 11	Jog Fwd	10 01	Ack
2X 12	Var Fwd	10 01	Ack
2X 13	Shuttle Fwd	10 01	Ack
20 20	Rewind	10 01	Ack
2X 21	Jog Rev	10 01	Ack
2X 22	Var Rev	10 01	Ack
2X 23	Shuttle Rev	10 01	Ack
20 30	Preroll	10 01	Ack
24 31	Cue up with Data	10 01	Ack
20 34	Sync Play	10 01	Ack
21 38	Prog Speed Play +	10 01	Ack
21 39	Prog Speed Play -	10 01	Ack
20 40	Preview	10 01	Ack
20 41	Review	10 01	Ack
20 42	Auto Edit	10 01	Ack
20 43	Outpoint Preview	10 01	Ack
2X 54	Anti-Clog Timer Disable	10 01	Ack
2X 55	Anti-Clog Timer Enable	10 01	Ack
20 60	Full EE Off	10 01	Ack
20 61	Full EE On	10 01	Ack
20 63	Select EE On	10 01	Ack
20 64	Edit Off	10 01	Ack
20 65	Edit On	10 01	Ack
20 6A	Freeze Off	10 01	Ack

20 6B	Freeze On	10 01	Ack
44 00	Timer-1 Preset	10 01	Ack
44 04	Time Code Preset	10 01	Ack
44 05	User Bit Preset	10 01	Ack
40 08	Timer-1 Reset	10 01	Ack
40 10	In Entry	10 01	Ack
40 11	Out Entry	10 01	Ack
40 12	Audio In Entry	10 01	Ack
40 13	Audio Out Entry	10 01	Ack
44 14	In Data Preset	10 01	Ack
44 15	Out Data Preset	10 01	Ack
44 16	Audio In Data Preset	10 01	Ack
44 17	Audio Out Data Preset	10 01	Ack
40 18	In + Shift	10 01	Ack
40 19	In - Shift	10 01	Ack
40 1A	Out + Shift	10 01	Ack
40 1B	Out - Shift	10 01	Ack
40 1C	Audio In + Shift	10 01	Ack
40 1D	Audio In - Shift	10 01	Ack
40 1E	Audio Out + Shift	10 01	Ack
40 1F	Audio Out - Shift	10 01	Ack
40 20	In Flag Reset	10 01	Ack
40 21	Out Flag Reset	10 01	Ack
40 22	Audio In Flag Reset	10 01	Ack
40 23	Audio Out Flag Reset	10 01	Ack
40 24	In Recall	10 01	Ack

40 25	Out Recall	10 01	Ack
40 26	Audio In Recall	10 01	Ack
40 27	Audio Out Recall	10 01	Ack
40 2D	Lost Lock Reset	10 01	Ack
4X 30	Edit Preset	10 01	Ack
44 31	Preroll time preset	10 01	Ack
41 32	Tape/Auto Select	10 01	Ack
41 33	Servo Ref Select	10 01	Ack
41 34	Head Select	10 01	Ack
41 35	Color Frame select	10 01	Ack
41 36	Timer Mode Select	10 01	Ack
41 37	Input Check	10 01	Ack
41 3A	Edit Field Select	10 01	Ack
41 3B	Freeze Mode Select	10 01	Ack
4X 3E	Record Inhibit	10 01	Ack
40 40	Auto Mode Off	10 01	Ack
40 41	Auto Mode On	10 01	Ack
40 42	Spot Erase Off	10 01	Ack
40 43	Spot Erase On	10 01	Ack
40 44	Audio Split Off	10 01	Ack
40 45	Audio Split On	10 01	Ack
4X 98	Output H Phase	10 01	Ack
4X 9B	Output Video Phase	10 01	Ack
4X A0	Audio Input Level	10 01	Ack
4X A1	Audio Output Level	10 01	Ack
4X A2	Audio Adv Level	10 01	Ack

4X A8	Audio Output Phase	10 01	Ack
4X A9	Audio Adv Out Phase	10 01	Ack
4X AA	Cross Fade Time Preset	10 01	Ack
4X B8	Local Key Map	10 01	Ack
42 F8	Still Off time	10 01	Ack
42 FA	Stby Off time	10 01	Ack
61 0A	TC Gen Sense	74 08	Gen Time Data
		79 09	Gen User Bits Data
		74 00	Timer-1 Data
		74 01	Timer-2 Data
		74 04	LTC Time Data
		74 05	User Bits (LTC) Data
61 0C	Current Time Sense	74 06	VITC Time Data
		74 07	User Bits (VITC) Data
		74 14	Corrected LTC Time Data
		74 15	Hold User Bits (LTC) Data
		74 16	Hold VITC Time Data
		74 17	Hold User Bits (VITC) Data
60 10	In Data Sense	74 10	In Data
60 11	Out Data Sense	74 11	Out Data
60 12	Audio In Data Sense	74 12	Audio In Data
60 13	Audio Out Data Sense	74 13	Audio Out Data
61 20	Status Sense	7X 20	Status Data
61 21	Extended VTR Status	7X 21	Extended Status Data
62 23	Signal Control Sense	7X 23	Signal Control Data

6X 28	Local Key Map Sense	7X 28	Local Key Map
61 2A	Head Meter Sense	7X 2A	Head Meter Data
60 2B	Remaining Time Sense	76 2B	Remaining Time
60 2E	Cmd Speed Sense	7X 2E	Cmd Speed Data
61 30	Edit Preset Sense	7X 30	Edit Preset Status
60 31	Preroll Time Sense	74 31	Preroll Time
60 36	Timer Mode Sense	71 36	Timer Mode Status
60 3E	Record Inhibit Sense	72 3E	Record Inhibit Status
60 52	DA Inp Emph Sense	71 52	DA Input Emphasis Data
60 53	DA PB Emph Sense	71 53	DA Playback Emphasis Data
60 58	DA Samp. Freq. Sense	71 58	DA Sampling Frequency Data
61 AA	Cross Fade Time Sense	7X AA	Cross Fade Time Data

Command Formats

00 0C Local Disable

Disables operation of the slave device from its control panel.

00 11 Device Type Request

Slave Responds with

12 11 Device Type

message, with 2 bytes of data:

Model	Data
BVU-800	10 00
BVW-10	2X 00
BVW-11	2X 02
BVW-15	2X 03
BVW-35	2X 10
BVW-40	2X 01

BVW-50	2X 30
BVW-60	2X 20
BVW-65	2X 21
BVW-95	2X 22
BVW-96	2X 23
BVW-70	2X 24
BVW-75	2X 25
BVW-D75	2X 46
BVW-9000	2X 47
PVW-2600	2X 40
PVW-2800	2X 41
BVW-35PM	20 18
BVW-65PM	20 29
BVW-95PM	20 2A
BVW-75PM	20 2D
BVW-85P	21 26
BVW-70S	21 2C
BVW-75S	21 2D
WBR-700	21 2D
DVR-2000	3X 10
DVR-2100	3X 11

Where X=0 for NTSC/PAL-M (525) models and 1 for PAL/SECAM models.

00 1D Local Enable

Enable operation of slave device from local panel according to the local enable map set by the "4X B8" Local Key Map command.

10 01 ACK

Slave sends this when it receives a command from Master.

11 12 NAK

When a communication error is detected, the slave sends this command with a "1" in the following position indicating the appropriate error condition:

7	6	5	4	3	2	1	0
Time Out	Framing Error	Overrun Error	Parity Error	X	Checksum Error	X	Undefined command

20 00 Stop

Slave stops current motion.

20 01 Play

Slave Starts to play from current location. When the "Sync play" mode is selected from the System menu on the slave, "Play" has the same effect as 20.34 "Sync Play".

20 02 Record

Slave begins recording. Exactly what happens depends on Auto Mode, record lockout, and edit presets.

20 04 Standby Off

Turns off standby mode. For VTR, this causes the machine to unthread in stop. Affects EE/Tape selection. Available only in Stop mode.

20 05 Standby On

Turns on standby mode. For VTR, this causes the machine to stay threaded when in stop. Affects EE/Tape selection.

20 0F Eject

When this command is received, the slave will eject the tape.

20.10 Fast Fwd

When this command is received, the slave device will run in fast forward mode. The speed depends on the VTR; for the DVR2000 series it is 50 x play speed.

2X.11 Jog Forward

2X.12 Var Forward

2X.13 Shuttle Forward

When these commands are received the slave device will move forward with the speed indicated by DATA-1 and DATA-2.

When only DATA-1 is given, the speed will be given by

```
Tape Speed = 10^((N/32)-2) x play speed.
```

where N is the value of DATA-1. Some sample values are:

Speed	Speed Data
0.1	32 (20H)
1.0	64 (40H)
2.9	79 (4FH)
48.7	118 (76H)

When a more precise speed value is required, then DATA-2 will be added. The speed formula for this case is

```
Tape Speed = 10^((N/32)-2) + N'/256*(10^(((N+1)/32)-2)-10^((N/32)-2))
```

where N is the value of DATA-1 and N' is the value of DATA-2.

auth - in more standard terms, the formula says that DATA-2 is used to linearly interpolate between the value given by N and that of N+1.

The maximum jog speed is set in the System:System menu. The maximum Var speed is 3X play speed. The maximum shuttle speed is 50X play speed.

auth - There is considerable controversy over the minimum speed. For a speed value of 0, the above formula with only DATA-1 gives 10^-2, or .01 x play speed. The standard states that when a speed between 0 and the minimum is given, the slave moves at minimum speed. In fact, many editors and control systems intend a "Shuttle 0" command (21 13 00) to pause the device and have it stop without disengaging. Devices which fail to do so will creep about 1 frame/second in this situation.

20 20 Rewind

When it receives this command, the slave runs in reverse at maximum speed: on the DVR2000, this is 50xplay speed.

2X 21 Jog Rev

2X 22 Var Rev

2X 23 Shuttle Rev

When receiving one of the above commands, the slave will start running in accordance with the speed data defined by DATA-1 and DATA-2. For the maximum and minimum speed see the 2X.12 Shuttle Fwd command.

20 30 Preroll

When this command is received the slave will search to the preroll position defined as the value obtained by subtracting the preroll time set by the 44.31 Preroll Time Preset command from the IN POINT data stored in the IN ENTRY memory by the 40.10 In Entry command.

24 31 Cue Up With Data

Cues the slave to the indicated time. Time is formatted as follows:

```
  data-1     data-2      data-3     data-4
| Frame | | Seconds | | minutes | | hours |
|  10 1 | |   10 1  | |   10 1  | | 10 1  |
```

auth - this is how time is represented in all commands and responses using a time code. The numbers indicate that the 10s value is stored in the high nibble and the 1s value in the low nibble. This is not to be confused with the 80-bit SMPTE timecode which is present in the analog timecode track on tape, or with the VITC timecode.

20 34 Sync Play

Prerolls the slave for the preset preroll time, then enters play mode.

21 38 Prog Speed Play +

21 39 Prog Speed Play -

These commands play back the slave device in steps of 0.1% within the range of +/- 25.5% of play speed. DATA-1 contains an 8-bit speed value. The deviation from nominal play speed is

```
Deviation(%)=0.2 x speed value
```

20 40 Preview

20 41 Review

20 42 Auto Edit

When one of these commands is received the slave goes into the indicated mode.

This is all the spec states here. What actually happens is determined by the edit presets, in and out point selections, and ee/tape settings. Basically, all three set the device to the preroll position, and run at play speed up to the in point. In Preview, the slave switches to EE mode at the in point, and out at the out point, simulating the edit without disturbing the recording media. In review, the slave simply continues to play at the in point and rolls to the outpoint. In Auto Edit, the channels indicated by the edit presets are put into record at the in point and the recording proceeds to the out point.

20 43 Outpoint Preview

Sends transport to preset out point if insert mode is preset.

2X 54 Anti-Clog Timer Disable

Disables the anti-clog timer. This timer is responsible for unthreading the tape upon timeout to save wear on the heads. If a system disables this timer, it should take responsibility for head wear avoidance itself.

2X 55 Anti-Clog Timer Enable

Enables anti-clog timer.

20 60 Full EE Off

Clears all channels from EE mode regardless of EDIT PRESET channels assigned by the 41.30 EDIT PRESET command. It takes the slave 5 frames to perform this operation after it receives the command.

20 61 Full EE On

Sets all channels to EE mode regardless of EDIT PRESET channels assigned by the 41.30 EDIT PRESET command. It takes the slave 5 frames to perform this operation after it receives the command.

20 63 Select EE On

This command sets only the preset channels assigned by the 41.30 Edit Preset command to EE mode. The EE mode is cleared by the 20.64 Edit Off command. It takes the slave 5 frames to perform this operation after receiving the command.

20 64 Edit Off

This command will stop recording without affecting the state of motion of the device. Any channels in record will come out in response to this command, after 5 frames of delay. This command also clears the Manual Edit Record mode and the Select EE mode.

20 65 Edit On

This command is used to actually initiate recording. When the device is playing, and the edit presets set by the 4X 30 Edit Preset command are in place, the preset channels will enter record a fixed delay after this command is received. The slave will enter Edit Rec mode at this point. It takes the slave 5 frames to enter Edit Rec after receiving this command.

20 6A Freeze Off

This command un-freezes the output of the device.

20 6A Freeze On

This command freezes the output of the device. There is usually a 2-5 frame delay associated with the actual freeze.

44 00 Timer-1 Preset

Sets the Timer-1 value to the time code indicated by DATA-1 thru DATA-4. For the time format see the 24.31 Cue Up With Data command.

44 04 Time Code Preset

Sets the Time Code Generator value to the time code indicated by DATA-1 thru DATA-4. The data format is as per the 24.31 Cue Up With Data command, with two additional bits to indicate Color Frame and Drop Frame mode as follows:

Data-1 Bit 7	CF	Data-1 Bit 6	DF

0	Off	0	Off
1	On	1	On

44 05 User Bit Preset

This command presets the value given by DATA-1 through DATA-4 to the User Bits of the Time Code Generator

DATA-1		**DATA-2**		**DATA-3**		**DATA-4**	
MSD	**LSD**	**MSD**	**LSD**	**MSD**	**LSD**	**MSD**	**LSD**
Bin Grp 2	Bin Grp 1	Bin Grp 4	Bin Grp 3	Bin Grp 6	Bin Grp 5	Bin Grp 8	Bin Grp 7

40 08 Timer-1 Reset

Clears Tape timer 1 to 0

40 10 In Entry

Sets the video in point to the value displayed on the slave. This is the value of the selected tape timer.

40 11 Out Entry

Sets the video out point to the value displayed on the slave. This is the value of the selected tape timer.

40 12 Audio In Entry

Sets the audio in point to the value displayed on the slave. This is the value of the selected tape timer.

40 13 Audio Out Entry

Sets the audio out point to the value displayed on the slave. This is the value of the selected tape timer.

44 14 In Data Preset

Set the Video In Point to the value indicated by DATA-1 through DATA-4. The time format is as per the 24.31 Cue Up With Data command.

44 15 Out Data Preset

Set the Video Out Point to the value indicated by DATA-1 thru DATA-4. The time format is as per the 24.31 Cue Up With Data command.

44 16 Audio In Data Preset

Set the Audio In Point to the value indicated by DATA-1 thru DATA-4. The time format is as per the 24.31 Cue Up With Data command.

44 17 Audio Out Data Preset

Set the Audio Out Point to the value indicated by DATA-1 thru DATA-4. The time format is as per the 24.31 Cue Up With Data command.

40 18 In + Shift

Increments the Video in point by one frame.

40 19 In - Shift

Decrements the Video in point by one frame.

40 1A Out + Shift

Increments the Video out point by one frame.

40 1B Out - Shift

Decrements the Video out point by one frame.

40 1C Audio In + Shift

Increments the Audio in point by one frame.

40 1D Audio In - Shift

Decrements the Audio in point by one frame.

40 1E Audio Out + Shift

Increments the Audio out point by one frame.

40 1F Audio Out - Shift

Decrements the Audio out point by one frame.

40 20 In Flag Reset

40 21 Out Flag Reset

40 22 Audio In Flag Reset

40 23 Audio Out Flag Reset

Turn off the In Entry, Out Entry Audio In, and Audio Out lamps.

auth - do these have any effect on the status bits?

40 24 In Recall

40 25 Out Recall

40 26 Audio In Recall

40 27 Audio Out Recall

Turn on the In Entry, Out Entry Audio In, and Audio Out lamps.

40 2D Lost Lock Reset

Resets (sets to 0) the Data-8/Bit-6 Lost Lock bit in the 7X.20 status data . The LOST LOCK status will be set when the servo is unlocked in the Play, Rec, or Edit mode.

4X 30 Edit Preset

This command is used for selecting the edit mode and selection of preset audio and video channels. These values are used by the Edit On and Edit Off commands.

Byte	7	6	5	4	3	2	1	0
DATA-1:	X	Insert	Assemble	Video	X	TC	A2 (Cue)	A1 (Cue)
DATA-2:	X	X	X	X	DA4	DA3	DA2	DA1

when the 41.30 command is used, the audio channels are set as per the table in the Edit:Setup menu. When the 42.30 command is used and Bit1 or Bit0 of Data-1 are "1", the Cue channel is selected.

44 31 Preroll time preset

Presets the preroll time given by DATA-1 thru DATA-4. Only the seconds digit int DATA-2 is used, and must be in the range of 0-59. The time format is as per the 24.31 Cue Up With Data command.

41 32 Tape/Auto Select

the Tape/EE mode is seleted by DATA-1 as follows:

```
 Data-1  Mode
-----------------
00   Auto ( Tape/EE )
01   Tape
FF   Follows the mode set in the STATE MAP on the SYSTEM menu in
Local.
```

41 33 Servo Ref Select

Selects the SERVO REFERENCE signal according to DATA-1:

```
 DATA-1  State
-----------------
00   Auto
01   External
02   Input
FF   Follows the mode set in REFERENCE on the SYSTEM:SYSTEM menu
in Local.
```

41 34 Head Select

auth - this command is not implemented by the DVR2000, but many other systems use it to to select playback/record heads. On the DVR2000 the selection is automatic.

41 35 Color Frame select

Sets the COLOR FRAME mode of the servo system according to DATA-1:

```
 DATA-1  State
-----------------
01      2 Field
02      4 Field
03      8 Field
FF      Follows the mode set in LOCK FIELD on the SYSTEM:SYSTEM
menu in Local.
```

41 36 Timer Mode Select

Selects the TIMER system used in AUTO mode, and for display of the IN ENTRY, OUT ENTRY, IN PRESET, OUT PRESET, PREROLL, and CUE UP WITH DATA, etc. according to DATA-1:

```
 DATA-1  Mode
-----------------
00      TIME CODE
01      TIMER-1
02      TIMER-2
FF      Follows the mode set in F6:CRNT TM on the HOME menu in
Local.
```

41 37 Input Check

Switches the INPUT CHECK mode on/off as per DATA-1:

```
 DATA-1 Mode
00    INPUT CHECK off
01    INPUT signals sent out VIDEO and MONITOR AUDIO OUTPUT
connectors.
```

41 3A Edit Field Select

Assigns the field on which to start the EDIT as per DATA-1:

```
DATA-1  Mode
-------------
00      Field1/Field2: The edit will start on the field the
command
        was received on
01      Field 1
02      Field 2
FF      Follows the mode set in F4:TIMING on the EDIT SETUP menu
in
        Local.
```

The following commands use the field selection:

- 20.60 Full EE Off
- 20.61 Full EE On
- 20.63 Select EE On
- 20.64 Edit Off
- 20.65 Edit On
- 4X.30 Edit Preset

41 3B Freeze Mode Select

Assigns the contents of the freeze picture to be field or frame as per DATA-1:

```
DATA-1  Mode
------------
00      Field-1 or Field-2 ( not fixed )
01      Field 1
10      Field 2
11      Frame
```

4X 3E Record Inhibit

40 40 Auto Mode Off

40 41 Auto Mode On

These commands switch the AUTO mode off and on.

40 42 Spot Erase Off

40 43 Spot Erase On

These commands switch the spot erase mode off and on.

40 44 Audio Split Off

40 45 Audio Split On

These commands switch the audio split mode off and on.

4X 98 Output H Phase

Sets the output Horizontal phase. The 40.98 command sets the H Phase to 0, whereas the 41.98 command sets the H phase according to

```
H Phase = DATA_1 * 148 nsec
```

where DATA_1 is interpreted as a twos complement binary number in the range of -127 to 127.

4X 9B Output Video Phase

Sets the output sync phase. The 40.9B command sets the output sync phase to 0 whereas the 41.9B command sets the output sync phase according to

```
Sync Phase = DATA_1 * 74 nsec
```

where DATA_1 is interpreted as a twos complement binary number in the range of -127 to 127.

4X A0 Audio Input Level

Controls the audio input level. The control mode is assigned by bit 7 of DATA-1. , and the channel to be controlled is assigned by bit 6 to bit 0 of DATA-1. When the 41 A0 command is received, the audio input levels of the channels assigned by DATA-1 will be set to their reference levels. When CMDLEN is not 1, the level data will be composed of two bytes per *assigned* channel in the order DA1, DA2,

DA3, DA4, CUE. The DATA-1 bits look like:

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Value	Mode	X	X	Cue	DA4	DA3	DA2	DA1

For Example, when the DATA-1 is 0x11, the command looks like:

DATA-1	**DATA-2**	**DATA-3**	**DATA-4**	**DATA-5**
11	DA1 LSB	DA1 MSB	CUE LSB	CUE MSB

The level data are interpreted as per the formula:

```
Input Level = 20 log [(Level Data)/(4000h)] (dB)
```

Which gives coverage over the range of +12dB to -infinity.

4X A1 Audio Output Level

Controls the audio output level. When the 41 A1 command is received, the audio output level of the channels selected in DATA-1 will be set to the reference level. Otherwise, it operates as per the 4X.A0 Audio input level command.

4X A2 Audio Adv Level

Controls the audio advance level. When the 41 A2 *fixed doc error here* command is received, the audio advance level of the channels selected by DATA-1 will be set to the reference level. Otherwise, it operates as per the 4X.A0 Audio input level command.

4X A8 Audio Output Phase

Sets the audio output phase. The 42.A8 command determines the audio output phase according to the 16-bit value encoded with the low byte in DATA-1 and the high byte in DATA-2. The values are interpreted as 2s complement binary, and currently have a range of +/- 80 samples. When the 40.A8 command is received, the slave will set the audio ouput phase to its nominal value.

4X A9 Audio Adv Out Phase

Sets the Advance audio output phase. The 42.A9 command determines the audio advance phase according to the 16-bit value encoded with the low byte in DATA-1 and the high byte in DATA-2. The values are interpreted as 2s complement binary, and currently have a range of +0/-50 samples. the 40.A9 will set the audio advance phase to its nominal value.

4X AA Cross Fade Time Preset

4X B8 Local Key Map

When the slave receives the 00.1D Local Enable command, the control panel may be used according to the local key map that was set by this command. When the slave receives the 00.0C Local Disable command all the keys, buttons, and adjustment controls on the control panel are disabled. The Eject button can always be used. If the slave receives the 41.B8 command, the local key map is preset by the block level in accordance with DATA-1. IF it receives the 4X.B8 command (X > 2) The local key map is preset by the Switch level.

Block Level switches:

```
       Bit 7 Bit 6 Bit 5  Bit 4    Bit 3   Bit 2   Bit 1   Bit 0
                          Tracking Monitor Audio   Video
Transport
                          Control  Control Control Control Control

"1": This function will be enabled when in remote
"0": This function will be disabled in remote.
```

When DATA-2 or more are added, control data with two bytes per each block assigned by DATA-1 are added following DATA-1.

At present the transport switches are defined as follows:

Bit	7	6	5	4	3	2	1	0
1st Byte	Execute	Preroll	Search		Rec	Play	Stop	Standby
2nd Byte						Var	Jog	Shuttle

None of the other blocks have any switches assigned, but rather operate as follows:

Video Control: Video phase and Sync phase can be adjusted on the system menu in remote mode.

Audio Control: Audio levels and output phase can be adjusted on the Audio:DA out menu in remote mode.

Monitor Control: the wfm monitor output selection on the system:wfm monitor menu and the montior level adjustments and monitor out selection on the system:audio monitor menu can be adjusted in remote mode.

Tracking Control: Tracking adjustments in the system:tracking menu can be made in remote mode.

42 F8 Still Off time

42 FA Standby Off time

61 0A TC Gen Sense

60 10 In Data Sense

60 11 Out Data Sense

60 12 Audio In Data Sense

60 13 Audio Out Data Sense

61 0C Current Time Sense

Requests the TIME DATA or USER BITS. Slave responds as per DATA-1:

```
      Bit 7   Bit 6   Bit 5   Bit 4   Bit 3   Bit 2   Bit 1   Bit
0
                      VITC    LTC   TIMER 2 TIMER 1   VITC   LTC
                        UB      UB                      TIME
TIME
```

That is to say that the response follows the bit set in DATA-1 as per the following table

Request	Response
DATA-1=01 LTC TIME	74.04: LTC TIME DATA
	74.14: CORRECTED LTC TIME DATA
DATA-1=02 VITC TIME	74.06: VITC TIME DATA
	74.16: HOLD VITC TIME DATA
DATA-1=04 TIMER-1	74.00: TIMER-1 DATA
DATA-1=08 TIMER-2	74.01: TIMER-2 DATA
DATA-1=10 LTC UB	74.05: UB (LTC) DATA
	74.15: HOLD UB (LTC) DATA
DATA-1=20 VITC UB	74.07: UB (VITC) DATA
	74.17: HOLD UB (VITC) DATA

Now, when requesting timecode, the results depend on the tape speed because at very low speeds (less than .25 play speed) it may not be possible to recover timecode. However, if VITC is present, that may be used instead. To automate this decision process, Sony has provided the special 61.0C.03 command which will return the best source of time code as per the table below. Note that when both LTC and VITC are not good you get back the 74.14 corrected LTC data. In this case, the time is actually the last good LTC time corrected by the tape timer.

Tape speed > .25

LTC Status	VITC Status	Return Data	Return code
OK	OK	LTC	74.04 XX XX XX XX
NG	OK	VITC	74.06 XX XX XX XX
OK	NG	LTC	74.04 XX XX XX XX
NG	NG	LTC(*)	74.14 XX XX XX XX

Tape speed < .25

LTC Status	VITC Status	Return Data	Return code
OK	OK	VITC	74.06 XX XX XX XX
NG	OK	VITC	74.06 XX XX XX XX
OK	NG	LTC	74.04 XX XX XX XX
NG	NG	LTC(*)	74.14 XX XX XX XX

61 20 Status Sense

When the slave receives a 61.20 Status Sense command, it will respond with a 74 20 Status Data response. The starting byte number and number of bytes requested are encoded in DATA-1, with the starting reg in the high nibble (bits 7-4) and the requested byte count in the low nibble (bits 3-0).

61 21 Extended VTR Status

62 23 Signal Control Sense

6X 28 Local Key Map Sense

61 2A Head Meter Sense

60 2B Remaining Time Sense

60 2E Cmd Speed Sense

61 30 Edit Preset Sense

60 31 Preroll Time Sense

60 36 Timer Mode Sense

60 3E Record Inhibit Sense

60 52 DA Inp Emph Sense

60 53 DA PB Emph Sense

60 58 DA Samp. Freq. Sense

61 AA Cross Fade Time Sense

74 00 Timer-1 Data

DATA-1 thru DATA-4 contain the timer-1 data formatted as per the 24.31 Cue up with data command

74 01 Timer-2 Data

DATA-1 thru DATA-4 contain the timer-2 data formatted as per the 24.31 Cue up with data command

74 04 LTC Time Data

DATA-1 thru DATA-4 contain the LTC data formatted as per the 24.31 Cue up with data command

74 05 User Bits (LTC) Data

DATA-1 thru DATA-4 contain the LTC User bits data formatted at per the 44.05 User Bit Preset command.

74 06 VITC Time Data

DATA-1 thru DATA-4 contain the VITC data formatted as per the 24.31 Cue up with data command

74 07 User Bits (VITC) Data

DATA-1 thru DATA-4 contain the VITC User bits data formatted at per the 44.05 User Bit Preset command.

74 08 Gen Time Data

DATA-1 thru DATA-4 contain the Time code generator data formatted as per the 24.31 Cue up with data command

79 09 Gen User Bits Data

DATA-1 thru DATA-4 contain the Time code generator User bits data formatted at per the 44.05 User Bit Preset command.

74 10 In Data

74 11 Out Data

74 12 Audio In Data

74 13 Audio Out Data

74 14 Corrected LTC Time Data

74 15 Hold User Bits (LTC) Data

74 16 Hold VITC Time Data

74 17 Hold User Bits (VITC) Data

74 20 Status Data

When the slave receives a 61.20 Status Sense command, the following data will be sent back in response according to the request:

Byte No.	7	6	5	4	3	2	1	0
0	X	X	Tape Out	Servo Ref Missing	X	X	X	Local
1	Standby	X	Stop	Eject	Rewind	Fast Fwd	Record	Play
2	Servo Lock	TSO Mode	Shuttle	Jog	Var	Tape Dir	Still	Cue Up
3	Auto Mode	Freeze On	X	CF Mode	A Out	A In	Out	In
4	Select EE	Full EE	X	Edit	Review	Auto Edit	Preview	Preroll
5	X	Insert	Assemble	Video	A4	A3	A2	A1
6	X	Lamp Still	Lamp Fwd	Lamp Rev	Srch LED 8	Srch LED 4	Srch LED 2	Srch LED 1
7	X	X	Aud Split	Sync Act	X	Spot Erase	X	In/Out
8	Buzzer	Lost lock	Near EOT	EOT	CF Lock	Svo Alarm	Sys Alarm	Rec Inhib
9	Fnc Abort	X	X	X	X	X	X	X

Description of Bits:

Data 0:

- Bit 5: Tape Unthreaded (Cassette Out) When the tape is threaded, this is 0. When it is completely unthreaded this is 1. When threading or unthreading, who knows what it is.
- Bit 4: Servo Ref Missing When servo reference is absent this is 1.
- Bit 0: Local When remote operation is inhibited by the "remote/local" switch

on the panel this is 1.

Data 1:

- Bit 7: Standby When the tape is threaded and the scanner is locked but the tape is stopped, this is 1.
- Bit 5: Stop When the machine is in full stop, this is 1. The thread state depends on the tape/ee and standby settings.
- Bit 4: Eject When the tape is ejecting this is 1.
- Bit 3: Rewind When the machine is in fast reverse this is 1.
- Bit 2: Fast Fwd When the machine is in fast forward this is 1.
- Bit 1: Record This bit goes from 0 to 1 some number of frames after the machine starts recording. For the DVR2000 we measured 5 frames. Others have varying delays on the record status.
- Bit 0: Play This bit goes from 0 to 1 some number of frames after the machine starts playing. For the DVR2000 we measured 5 frames. Others have varying delays on the play status.

Data 2:

- Bit 7: Servo Lock 1 indicates servos are locked. This is a necessary condition for an edit to occur correctly.
- Bit 6: TSO Mode Bit is 1 in tape speed override: in this mode, audio and video are still locked though speed is off play speed by +/- up to 15%.
- Bit 5: Shuttle
- Bit 4: Jog
- Bit 3: Var
- Bit 2: Tape Dir
- Bit 1: Still
- Bit 0: Cue Up

Data 3:

- Bit 7: Auto Mode
- Bit 6: Freeze On
- Bit 4: CF Mode
- Bit 3: A Out
- Bit 2: A In

- Bit 1: Out
- Bit 0: In

Data 4:

- Bit 7: Select EE
- Bit 6: Full EE
- Bit 4: Edit
- Bit 3: Review
- Bit 2: Auto Edit
- Bit 1: Preview
- Bit 0: Preroll

Data 5:

- Bit 6: Insert
- Bit 5: Assemble
- Bit 4: Video
- Bit 3: A4
- Bit 2: A3
- Bit 1: A2
- Bit 0: A1

Data 6:

- Bit 6: Lamp Still
- Bit 5: Lamp Fwd
- Bit 4: Lamp Rev
- Bit 3: Srch LED 8
- Bit 2: Srch LED 4
- Bit 1: Srch LED 2
- Bit 0: Srch LED 1

Data 7:

- Bit 5: Aud Split

- Bit 4: Sync Act
- Bit 2: Spot Erase
- Bit 0: In/Out

Data 8:

- Bit 7: Buzzer
- Bit 6: Lost lock This bit is controled by the Lost Lock Reset command. It is set when the servos are unlocked in the PLAY, REC, or EDIT modes.
- Bit 5: Near EOT
- Bit 4: EOT
- Bit 3: CF Lock
- Bit 2: Svo Alarm
- Bit 1: Sys Alarm
- Bit 0: Rec Inhib

Data 9:

- Bit 7: Fnc Abort

---

auth - here's a code snippet for you c-heads.

```
char *StatusBitStrings[][8] ={
        // Data 0
        {"          ","           ","Unthread  ","Svo Ref    ",
         "           ","           ","           ","Local      "},
        // Data 1
        {"Standby    ","           ","Stop       ","Eject      ",
         "Rewind     ","Fast Fwd   ","Record     ","Play       "},
        // Data 2
        {"Servo Lock","TSO Mode   ","Shuttle    ","Jog        ",
         "Variable   ","Tape Rev   ","Still      ","Cue        "},
        // Data 3
        {"Auto Mode ","Freeze On ","           ","CF Mode    ",
         "A Out      ","A In       ","Out        ","In         "},
        // Data 4
        {"Select EE ","Full EE    ","           ","Edit       ",
         "Review     ","Auto Edit ","Preview    ","Preroll    "},
        // Data 5
        {"           ","Insert     ","Assemble   ","Video      ",
         "A4         ","A3         ","A2         ","A1         "},
        // Data 6
        {"           ","Lamp Still","Lamp Fwd   ","Lamp Rev   ",
         "Srch LED 8","Srch LED 4","Srch LED 2","Srch LED 1"},
        // Data 7
        {"           ","           ","Aud Split ","Sync Act   ",
         "           ","Spot Erase","           ","In/Out     "},
        // Data 8
```

```
{"Buzzer    ","Lost lock ","Near EOT  ","EOT       ",
 "CF Lock   ","Svo Alarm ","Sys Alarm ","Rec Inhib "},
// Data 9
{"Fnc Abort ","          ","          ","          ",
 "          ","          ","          ","          "},};
```

The Status bits communicate much about the progress of a motion command. The precise timing of each signal varies almost from machine to machine, and many edit controllers expect certain timing behaviour of these signals. Herein lie many of the problems associated with edit controllers, edit timing, and just plain wacky transport behavior.

7X 21 Extended Status Data

7X 23 Signal Control Data

7X 28 Local Key Map

7X 2A Head Meter Data

76 2B Remaining Time

7X 2E Cmd Speed Data

7X 30 Edit Preset Status

74 31 Preroll Time

71 36 Timer Mode Status

72 3E Record Inhibi Status

71 52 DA Input Emphasis Data

71 53 DA Playback Emphasis Data

71 58 DA Sampling Frequency Data

7X AA Cross Fade Time Data

Rick Davis<rld@well.com>

Appendice B Strutture dati AVI

Si riporta qui di seguito la descrizione delle strutture dati AVISTREAMINFO e WAVEFORMATEX tratte dalla "MSDN Library" [66].

B.1 AVISTREAMINFO

The **AVISTREAMINFO** structure contains information for a single stream.

```
typedef struct {
    DWORD fccType;
    DWORD fccHandler;
    DWORD dwFlags;
    DWORD dwCaps;
    WORD  wPriority;
    WORD  wLanguage;
    DWORD dwScale;
    DWORD dwRate;
    DWORD dwStart;
    DWORD dwLength;
    DWORD dwInitialFrames;
    DWORD dwSuggestedBufferSize;
    DWORD dwQuality;
    DWORD dwSampleSize;
    RECT  rcFrame;
    DWORD dwEditCount;
    DWORD dwFormatChangeCount;
    char  szName[64];
} AVISTREAMINFO;
```

Members

fccType

Four-character code indicating the stream type. The following constants have been defined for the data commonly found in AVI streams:

Constant	Description
StreamtypeAUDIO	Indicates an audio stream.
StreamtypeMIDI	Indicates a MIDI stream.
StreamtypeTEXT	Indicates a text stream.
StreamtypeVIDEO	Indicates a video stream.

fccHandler

Four-character code of the compressor handler that will compress this video stream when it is saved (for example, **mmioFOURCC**('M','S','V','C')). This member is not used for audio streams.

dwFlags

Applicable flags for the stream. The bits in the high-order word of these flags are

specific to the type of data contained in the stream. The following flags are defined:

AVISTREAMINFO_DISABLED
Indicates this stream should be rendered when explicitly enabled by the user.

AVISTREAMINFO_FORMATCHANGES
Indicates this video stream contains palette changes. This flag warns the playback software that it will need to animate the palette.

dwCaps
Capability flags; currently unused.

wPriority
Priority of the stream.

wLanguage
Language of the stream.

dwScale
Time scale applicable for the stream. Dividing dwRate by dwScale gives the playback rate in number of samples per second.
For video streams, this rate should be the frame rate. For audio streams, this rate should correspond to the audio block size (the nBlockAlign member of the WAVEFORMAT or PCMWAVEFORMAT structure), which for PCM (Pulse Code Modulation) audio reduces to the sample rate.

dwRate
Rate in an integer format. To obtain the rate in samples per second, divide this value by the value in dwScale.

dwStart
Sample number of the first frame of the AVI file. The units are defined by dwRate and dwScale. Normally, this is zero, but it can specify a delay time for a stream that does not start concurrently with the file.
The 1.0 release of the AVI tools does not support a nonzero starting time.

dwLength
Length of this stream. The units are defined by dwRate and dwScale.

dwInitialFrames
Audio skew. This member specifies how much to skew the audio data ahead of the video frames in interleaved files. Typically, this is about 0.75 seconds.

dwSuggestedBufferSize
Recommended buffer size, in bytes, for the stream. Typically, this member contains a value corresponding to the largest chunk in the stream. Using the correct buffer size makes playback more efficient. Use zero if you do not know the correct buffer size.

dwQuality
Quality indicator of the video data in the stream. Quality is represented as a number between 0 and 10,000. For compressed data, this typically represents the value of the quality parameter passed to the compression software. If set to – 1, drivers use the default quality value.

dwSampleSize
Size, in bytes, of a single data sample. If the value of this member is zero, the samples can vary in size and each data sample (such as a video frame) must be in a separate chunk. A nonzero value indicates that multiple samples of data can be grouped into a single chunk within the file.
For video streams, this number is typically zero, although it can be nonzero if all video frames are the same size. For audio streams, this number should be the same

as the nBlockAlign member of the WAVEFORMAT or WAVEFORMATEX structure describing the audio.

rcFrame

Dimensions of the video destination rectangle. The values represent the coordinates of upper left corner, the height, and the width of the rectangle.

dwEditCount

Number of times the stream has been edited. The stream handler maintains this count.

dwFormatChangeCount

Number of times the stream format has changed. The stream handler maintains this count.

szName

Null-terminated string containing a description of the stream.

Requirements

Windows NT/2000: Requires Windows NT 3.1 or later.

Windows 95/98: Requires Windows 95 or later.

Windows CE: Unsupported.

Header: Declared in vfw.h.

Unicode: Defined as Unicode and ANSI structures.

See Also

AVIFile Functions and Macros Overview, AVIFile Structures, **mmioFOURCC**, **PCMWAVEFORMAT**, **WAVEFORMAT**, **WAVEFORMATEX**

B.2 WAVEFORMATEX

The **WAVEFORMATEX** structure defines the format of waveform-audio data. Only format information common to all waveform-audio data formats is included in this structure. For formats that require additional information, this structure is included as the first member in another structure, along with the additional information.

```
typedef struct {
    WORD  wFormatTag;
    WORD  nChannels;
    DWORD nSamplesPerSec;
    DWORD nAvgBytesPerSec;
    WORD  nBlockAlign;
    WORD  wBitsPerSample;
    WORD  cbSize;
} WAVEFORMATEX;
```

Members

wFormatTag
Waveform-audio format type. Format tags are registered with Microsoft Corporation for many compression algorithms. A complete list of format tags can be found in the MMREG.H header file.

nChannels
Number of channels in the waveform-audio data. Monaural data uses one channel and stereo data uses two channels.

nSamplesPerSec
Sample rate, in samples per second (hertz), that each channel should be played or recorded. If **wFormatTag** is WAVE_FORMAT_PCM, then common values for **nSamplesPerSec** are 8.0 kHz, 11.025 kHz, 22.05 kHz, and 44.1 kHz. For non-PCM formats, this member must be computed according to the manufacturer's specification of the format tag.

nAvgBytesPerSec
Required average data-transfer rate, in bytes per second, for the format tag. If **wFormatTag** is WAVE_FORMAT_PCM, **nAvgBytesPerSec** should be equal to the product of **nSamplesPerSec** and **nBlockAlign**. For non-PCM formats, this member must be computed according to the manufacturer's specification of the format tag.
Playback and record software can estimate buffer sizes by using the **nAvgBytesPerSec** member.

nBlockAlign
Block alignment, in bytes. The block alignment is the minimum atomic unit of data for the **wFormatTag** format type. If **wFormatTag** is WAVE_FORMAT_PCM, **nBlockAlign** should be equal to the product of **nChannels** and **wBitsPerSample** divided by 8 (bits per byte). For non-PCM formats, this member must be computed according to the manufacturer's specification of the format tag.
Playback and record software must process a multiple of **nBlockAlign** bytes of data at a time. Data written and read from a device must always start at the beginning of a block. For example, it is illegal to start playback of PCM data in

the middle of a sample (that is, on a non-block-aligned boundary).

wBitsPerSample

Bits per sample for the **wFormatTag** format type. If **wFormatTag** is WAVE_FORMAT_PCM, then **wBitsPerSample** should be equal to 8 or 16. For non-PCM formats, this member must be set according to the manufacturer's specification of the format tag. Note that some compression schemes cannot define a value for **wBitsPerSample**, so this member can be zero.

cbSize

Size, in bytes, of extra format information appended to the end of the **WAVEFORMATEX** structure. This information can be used by non-PCM formats to store extra attributes for the **wFormatTag**. If no extra information is required by the **wFormatTag**, this member must be set to zero. Note that for WAVE_FORMAT_PCM formats (and only WAVE_FORMAT_PCM formats), this member is ignored.

Remarks

An example of a format that uses extra information is the Microsoft Adaptive Delta Pulse Code Modulation (MS-ADPCM) format. The **wFormatTag** for MS-ADPCM is WAVE_FORMAT_ADPCM. The **cbSize** member will typically be set to 32. The extra information stored for WAVE_FORMAT_ADPCM is coefficient pairs required for encoding and decoding the waveform-audio data.

Requirements

Windows NT/2000: Requires Windows NT 3.1 or later.

Windows 95/98: Requires Windows 95 or later.

Windows CE: Unsupported.

Header: Declared in vfw.h.

Unicode: Defined as Unicode and ANSI structures.

See Also

AVIFile Functions and Macros Overview, AVIFile Structures, **mmioFOURCC**, **PCMWAVEFORMAT**, **WAVEFORMAT**, **WAVEFORMATEX**

Bibliografia

[1] "Betacam PALsite - The Format"; disponibile presso http://betacam.palsite.com/format.html

[2] E. Fleischman; "Advanced Streaming Format (ASF) Specification"; 26 febbraio 1998; Internet Engineering Task Force Internet-Draft, draft-fleischman-asf-01; disponibile presso http://www.globecom.net/ietf/draft/draft-fleischman-asf-01.txt

[3] T. Berners-Lee, L. Masinter, M. McCahill; "Uniform Resource Locators (URL)"; RFC-1738; dicembre 1994

[4] Electronic Industries Association; "Electrical characteristics of balanced voltage digital interface circuits"; RS-422A

[5] J. Oostergaard; "The Software-RAID HOWTO"; 19 gennaio 2000; disponibile presso http://www.linux.org/docs/ldp/howto/Software-RAID-HOWTO.html

[6] Jupiter Media Matrix; "RealNetworks tops Microsoft for media-player users"; novembre 2000; disponibile presso http://yahoofin.cnet.com/news/0-1005-200-4561161.html?tag=rltdnws

[7] Jupiter Media Matrix; aprile 2001; disponibile presso http://www.jmm.com/xp/jmm/press/2001/pr_040301.xml

[8] R. McMordie ; "A brief history of the development of Teletext and Viewdata"; 3 ottobre 1998; disponibile presso http://www.mcmordie.demon.co.uk/computing/develhist.html

[9] RDS Forum; "What is RDS?"; 3 gennaio 2000; disponibile presso http://www.rds.org.uk/rds98/rds98.htm

[10] CENELEC; "Specifications of the radio data system; EN 50067"; European Committee for Electrotechnical Standardization; Brussels, Belgium; 1992

[11] Robbie W3RSM; "CTCSS, PL, Tone Squelch, and other Necessary Evils"; disponibile presso http://www.eecis.udel.edu/~dra/pl.html

[12] Radiocommunications Agency; "CODE OF PRACTICE - Continuous tone controlled signalling system (CTCSS) for use in the Land Mobile Services"; MPT 1306; Londra; ottobre 1996

[13] Teltronic; "Repeater RP-30S - The professional solution for mobile communication Appropiate for voice and data communication";

Spagna/Brasile; disponibile presso
http://www.teltronic.es/pdfs_pwpt/rp30_ing.pdf

[14] F. Pollastri; Istituto Elettrotecnico Nazionale; disponibile presso
http://toi.iriti.cnr.it/ist/it/iensrc.html

[15] ISO R226 (1961); "Normal equal-loudness contours for pure tones and normal threshold of hearing under free-field listening conditions"; International Standards Organization, Ginevra, Svizzera; http://www.iso.org

[16] K. D. Donohue; "Frequency Response of the Ear"; University of Kentucky, KY, USA; disponibile presso
http://www.engr.uky.edu/~donohue/audio/fsear.html

[17] ISO/IEC 11172-3; "Coding of Moving pictures and associated audio for digital storage media at up to 1.5 Mbit/s - Audio Part"; International Standard Organization; 1993

[18] K. H. Brandenburg, G. Stoll et al.; "ISO-MPEG-1 Audio: A Generic Standard for Coding of High Quality Digital Audio"; Journal of the Audio Engineering Society; ottobre 1994; Vol. 42, No. 10, pp. 780 - 792

[19] P. Noll; "MPEG Digital Audio Coding"; IEEE Signal Processing Magazine; settembre 1997; pp. 59-81

[20] S. Shlien; "Guide to MPEG-1 Audio Standard"; IEEE Transactions on Broadcasting; dicembre 1994; Vol. 40, No. 4, pp. 206-218

[21] D. Pan; "A tutorial on mpeg audio compression"; IEEE Multimedia Journal, Summer 1995; Vol. 2, No. 7, pp 60-74

[22] B. Scharf, "Critical Bands," Foundations of Modern Auditory Theory , J. Tobias, pp 159-202, Academic Press, New York and London, 1970

[23] J.D.Johnston, "Estimation of Perceptual Entropy Using Noise Masking Criteria," Proc. of the Int. Conf. IEEE ASSP, pp 2524-2527, 1988

[24] J.D.Johnston, "Transform Coding of Audio Signals Using Perceptual Noise Criteria," IEEE Journal on Selected Areas in Communications, vol. 6, pp 314-323, febbraio 1988

[25] K.Brandenburg, "OCF - A New Coding Algorithm for High Quality Sound Signals," Proc. of the Int. Conf. IEEE ASSP, pp. 141-144, 1987

[26] D.Wiese and G. Stoll, "Bitrate Reduction of High Quality Audio Signals by Modeling the Ears' Masking Thresholds," 89th AES-Convention, preprint 2970, Los Angeles 1990

[27] P. Prandoni, M. Vetterli; "R/D Optimal Data Hiding"; Ecole Polytechnique de Lausanne, Losanna, Svizzera

[28] "Basics about MPEG Perceptual Audio Coding"; Fraunhofer Institut Integrierte Schaltungen; disponibile presso http://www.iis.fhg.de/amm/techinf/basics.html

[29] Hong-Meng Liem, Woon-Seng Gan; "Headphone Equalization Using Analog Devices DSP"; febbraio 2000; Nanyang Technological University, Singapore

[30] SONY; "Operating Instructions - PUW-2800P"; 1991

[31] Audio Excel; "Sound Card AV-310 (CMI8330) User Manual"

[32] "Sound Conductor 16 PnP (ESS 1868) User Manual"

[33] Letizia Lo Presti, Fabio Neri; "L'analisi dei segnali"; 1992; CLUT; Torino

[34] Sennheiser; “Technical Specifications e602”; Germania

[35] "Pinnacle Reeltime NITRO Specifications"; disponibile presso http://www.pinnaclesys.de/asia/rtn_specs.asp

[36] NCH Swift Sound; "NCH Tone Generator"; (c) 1998, 1999; disponibile presso http://www.nch.com.au

[37] Mike Zwingl OE3MZC; "Single tone decoder"; disponibile presso http://www.qsl.net/oe3mzc/ne567.gif

[38] Philips Semiconductors; "Tone decoder/phase-locked loop NE/SE567"; 15 aprile 1992; disponibile presso http://www-us6.semiconductors.com/pip/ne567d

[39] E. O. Brigham, "The Fast Fourier Transform and its Applications"; Prentice-Hall; Englewood Cliffs, NJ; 1988

[40] Kurt L. Kosbar; "Fourier Transforms"; 21 aprile 1998; Electrical Engineering Department; University of Missouri - Rolla, USA; disponibile presso http://www.siglab.ece.umr.edu/ee301/dsp/fourier/fourier.html

[41] W. R. Bennett; "Spectrum of quantized signals"; B.S.T.J.; vol 27, pp 446-472; luglio 1948

[42] Cambridge University Press; "NUMERICAL RECIPES IN C: THE ART OF SCIENTIFIC PROGRAMMING"; ISBN 0-521-43108-5; pp 501-504

[43] Kurt L. Kosbar; "Scary Example"; 16 febbraio 1998; Electrical Engineering Department; University of Missouri - Rolla; disponibile presso http://www.siglab.ece.umr.edu/ee301/dsp/fourier/scary.html

[44] "Windows" disponibile presso http://astronomy.swin.edu.au/pbourke/analysis/windows/

[45] Kurt L. Kosbar; "Weighting Windows"; 25 febbraio 1998; Electrical Engineering Department; University of Missouri - Rolla; disponibile presso http://www.siglab.ece.umr.edu/ee301/dsp/fourier/windows.html

[46] "Bartlett Window"; disponibile presso http://www.cg.tuwien.ac.at/studentwork/CESCG99/TTheussl/node6.html

[47] Cambridge University Press; "NUMERICAL RECIPES IN C: THE ART OF SCIENTIFIC PROGRAMMING"; ISBN 0-521-43108-5; pp 549-558

[48] Cambridge University Press; "NUMERICAL RECIPES IN C: THE ART OF SCIENTIFIC PROGRAMMING"; ISBN 0-521-43108-5; pp 506-508

[49] Cambridge University Press; "NUMERICAL RECIPES IN C: THE ART OF SCIENTIFIC PROGRAMMING"; ISBN 0-521-43108-5; pp 512-514

[50] J. W. Cooley, J. W. Tukey; "An algorithm for machine computation of comlpex Fourier series"; Math. Comp.; vol 19, pp 297-381; aprile 1965

[51] R. Davies; "Sony 9-Pin Remote Protocol"; 1996; disponibile presso http://www.belle-nuit.com/archives/9pin.html

[52] D. S. Lawyer; "Serial HOWTO"; disponibile presso http://www.linux.org/docs/ldp/howto/Serial-HOWTO.html; febbraio 2001

[53] R. Saikkonen; "Linux I/O Port Programming mini-HOWTO"; 13 dicembre 2000; disponibile presso http://www.linux.org/docs/ldp/howto/mini/IO-Port-Programming.html

[54] P. H. Baumann; "The Linux Serial Programming HOWTO"; disponibile presso http://www.linux.org/docs/ldp/howto/Serial-Programming-HOWTO.html; 22 gennaio 1998

[55] B. Ward; "The Linux Kernel HOWTO"; 28 marzo 2001; disponibile presso http://www.linux.org/docs/ldp/howto/Kernel-HOWTO.html

[56] J. Tranter; "The Linux Sound HOWTO"; 24 marzo 1999; disponibile presso http://www.linux.org/docs/ldp/howto/Sound-HOWTO.html

[57] J. Tranter; "Open Sound System - Programmer's Guide - ver 1.11"; 7 novembre 2000; 4Front Technologies; disponibile presso http://www.opensound.com/pguide; pagina 9

[58] H. Savolainen; "Open Sound System - Audio Programming"; 1996; disponibile presso http://www.opensound.com/pguide/audio.html

[59] N. Jayant, P. Noll; "Digital Coding of Waveforms"; Englewood Cliffs, NJ, USA; Prentice Hall, 1984

[60] P. F. Swaszek; "Quantization"; New York, NY, USA; Van Nostrand Reinhold; 1985

[61] J. F. McGowan; "AVI Overview"; 22 luglio 1999; disponibile presso http://www.rahul.net/jfm/avi.html

[62] "MIDI - a whatis definition"; disponibile presso http://whatis.techtarget.com/definition/0,,sid9_gci212572,00.html; 16 gennaio 2001

[63] E. Fleischman; "WAVE and AVI Codec Registries"; RFC-2361; giugno 1998

[64] J. Salsman, H. Alvestrand; "The Audio/L16 MIME content type"; RFC-2586; maggio 1999

[65] H. Schulzrinne; "RTP Profile for Audio and Video Conferences with Minimal Control"; RFC-1890; gennaio 1996

[66] “msdn Library” distribuita a pagamento su CD-ROM, o disponibile a http://msdn.microsoft.com

[67] "Fletcher-Munson Equal Loudness Curves"; disponibile presso://www.allchurchsound.com/index.html?ACS/edart/fmelc.html

[68] Hamster; "Avifile"; versione 0.6.0; 31 agosto 2001; http://avifile.sf.net/